Žilinská univerzita v Žiline
Fakulta riadenia a informatiky

Ľuboš Takáč, Ing.

Autoreferát dizertačnej práce

Spracovanie dát v rozsiahlych databázach

na získanie akademického titulu **„philosophiae doctor“ (v skratke PhD.)**
v študijnom programe doktorandského štúdia
aplikovaná informatika

v študijnom odbore:
9.2.9 aplikovaná informatika

Žilina, Apríl, 2013

Dizertačná práca bola vypracovaná v dennej forme doktorandského štúdia **na katedre Informatiky, Fakulte riadenia a informatiky Žilinskej univerzity v Žiline.**

Predkladateľ: **Ing. Ľuboš Takáč**
Žilinská univerzita v Žiline
Fakulta riadenia a informatiky
Katedra Informatiky

Školiteľ: **Doc. Ing. Michal Zábovský, PhD.**
Žilinská univerzita v Žiline
Fakulta riadenia a informatiky
Katedra Informatiky

Oponenti:

Autoreferát bol rozoslaný dňa: .....

Obhajoba dizertačnej práce sa koná dňa ..... o **.....** h. pred komisiou pre obhajobu dizertačnej práce schválenou odborovou komisiou v študijnom odbore **9.2.9 aplikovaná informatika**, **v študijnom programe aplikovaná informatika**, vymenovanou dekanom Fakulty riadenia a informatiky Žilinskej univerzity v Žiline dňa .....

prof. Ing. Martin Klimo, PhD.
predseda odborovej komisie
študijného programu **aplikovaná informatika**
v študijnom odbore **9.2.9 aplikovaná informatika**
Fakulta riadenia a informatiky
Žilinská univerzita
Univerzitná 8215/1
010 26 Žilina

OBSAH

1 Úvod

Databázové systémy (DBS) sú neoddeliteľnou súčasťou takmer všetkých informačných systémov a väčších softvérových aplikácii. Môžeme sa s nimi stretnúť vo väčšine firiem, ale aj v štátnej sfére a inde. Slúžia nám na manipuláciu a uchovávanie väčšieho množstva logicky prepojených dát. Potrebnosť databáz a technologický pokrok dohnal tieto systémy na takú úroveň, že sme schopní uchovávať a manipulovať s obrovským množstvom dát (rádovo giga, tera, ale aj peta bajty), lenže už nie sme schopní len tak jednoducho sa v nich orientovať a využívať ich potenciál. Ide o takzvané zahltenie dátami. Správne využitie a spracovanie existujúcich dát môže pozitívne ovplyvniť zisk a chod firmy alebo byť konkurenčnou výhodou firmy na trhu. S narastajúcim počtom dostupných dát je nevyhnutné vyvíjať aj techniky a metódy ich spracovania tak, aby boli pochopiteľnejšie pre človeka a aby sa v nich

dokázal jednoduchšie orientovať. To je pohľad z užívateľského hľadiska a potreba využitia už existujúcich dát a dolovanie informácií z nich. Problém vzniká tiež na strane druhej, u databázových architektoch, vývojároch a ľudí čo navrhujú, vyvíjajú prípadne modifikujú takéto úložiská dát, alebo databázy. Samotný návrh veľkej databázy nie je triviálna záležitosť. Ešte zložitejšie sú úpravy nevyhnutné pre vývoj a prispôsobenie sa novým požiadavkám systému. Problém skúmania dizertačnej práce sa dá definovať nasledovne:

"Ako pochopiť rozsiahlu relačnú databázu do takej miery, aby sme ju vedeli efektívne používať, upravovať a orientovať sa v nej." [1]

1.1 Motivácia

Hlavnou motiváciou na riešenie tohto problému je moja osobná skúsenosť, keď som vo firme ako programátor potreboval často používať databázu, ktorá bola enormne veľká a bolo len málo ľudí čo ju poznalo tak, aby mi vedeli poradiť čo v nej kde nájdem. Najprv som si myslel, že je to len problém firmy a nevedia si urobiť v tom poriadok, ale keď som sa informoval od známych pracujúcich v rovnakej oblasti, bolo mi potvrdené, že to je bežná prax. Bežne dostupné proprietárne nástroje od dodávateľov DBS (MySQL Workbench, HeidiSQL, EMS SQL Manager for Oracle, IBM Data Studio, PGAdmin, a iné) a tiež komerčné (Bellman data profiler, Polaris Interactive DB Visualization) sú nepostačujúce pre databázy zložité z hľadiska štruktúry dátového modelu. V dostupnej literatúre [11-27], ktorej je k tejto problematike málo, nie je tento problém dostatočne vyriešený.

1.2 Ciele práce

V práci sú popísané aktuálne poznatky všeobecne o spracovaní dát a hlavne relačných databáz (RDB), ich použitie a ich nedostatky. Za tú dobu čo boli vyvíjané paralelne aj s inými informačnými technológiami a hardvérom sa dostali do takého stavu, že už nie je problém dáta zhromažďovať, robiť operácie nad nimi, rýchlo ich sprístupniť v krátkom čase na celom svete, ale pochopiť ich a orientovať sa v nich. Technológia nás predbehla v tomto smere a bez zmeny prístupu k týmto dátam a informáciám sa v nich budeme čoraz viacej strácať. Súčasné veľké objemy dát si vyžadujú nové metódy ich spracovania a používania. Bez toho aby sme sa v dátach vedeli orientovať a poznali vzťahy medzi nimi, sú nám zbytočné a nevyužijeme nikdy plne ich potenciál a informácie v nich ukryté.
V práci sa venujem hlavne možnostiam spracovania veľkých objemov dát a následne ich efektívneho pochopenia človekom pri danej forme spracovania a podania informácie. Takýto problém tiež nastáva pri snahe o pochopenie rozsiahlej relačnej databázy (RDB). Stovky tabuliek a vzťahov medzi nimi sa nedajú chápať naraz a ťažko odhadovať čo je podstatné a kde hľadať relevantné dáta a informácie. V práci riešim sémantika RDB a možnosti ako uľahčiť používateľom používanie RDB (najmä rozsiahlych), v ktorých sa ťažko hľadajú informácie a kde sa zle orientuje. Výsledky práce sú určené pre odborníkov pracujúcich s RDB, databázovým architektom pri údržbe a modifikácií existujúcich databáz, programátorom pracujúcich s RDB, aby sa v nich mohli rýchlo a efektívne orientovať, tiež databázovým špecialistom, ktorí navrhujú, vyvíjajú a optimalizujú RDB, ale aj laikom, ktorí chcú v RDB informácie vyhľadávať bez toho aby, ovládali technológie RDB systémov.

Prvým cieľom práce je analýza súčasného stavu spracovania dát. Preskúmanie a popis celého procesu spracovania dát (získanie, extrakcia, transformácia, použitie) a metód, ktoré sa aktuálne na spracovanie dát používajú, ich výhody a nedostatky. V práci je popísaný problém spracovania rozsiahlych dát a prístupov riešenia tohto problému. Zameral som sa hlavne na relačné databázy a získavanie informácií a znalostí z nich.

Druhým, hlavným cieľom práce je vytvorenie metodiky, postupu, algoritmu, alebo skupiny algoritmov, ktoré by umožňovali analyzovať rozsiahlu relačnú databázu, jej štruktúru, vybrať dôležité časti a vzťahy, ktoré by pomohli jednotlivcovi pochopiť takýto systém v čo najrýchlejšom čase a vyhľadať v ňom potrebné informácie a vzťahy s minimálnym rizikom chýb. Tieto metódy by mali byť všeobecne použiteľné a nezávislé na databázovom a operačnom systéme.

1.3 Rozsiahle DB, (Very Large DataBases)

Pod pojmom rozsiahle DB (VLDB) sa myslia databázy enormnej veľkosti, alebo zložitosti [1, 2, 16]. V súčasnosti DB s miliónmi záznamov, rádovo stovkami až tisíckami tabuliek a giga až terabajtami dát. Tento pojem nemá jednoznačnú definíciu. Za rozsiahlu DB sa môže považovať aj databáza zložitého modelu, alebo s veľkým počtom súčasne pracujúcich užívateľov.

VLDB problémy by sme mohli rozdeliť do dvoch základných typov:

- problémy spojené so štruktúrou (schéma DB, dátové modelovanie),
- problémy s dátami samotnými a získavaním informácii z nich (data mining).

Pri relačných databázových systémoch nám popisuje štruktúru a vzťahy relačný databázový model. Už ten samotný tvorí pridanú hodnotu databázového systému. Ak je dobre navrhnutý, je nositeľom skrytej informácie, čo si nie všetci uvedomujú. Pri veľmi veľkých RDB je problém nie len s návrhom tohto modelu, ale neskôr najmä s jeho pochopením a orientovaním sa v ňom. Práca s RDB schémou, ktorá má rádovo stovky tabuliek a vzťahov medzi nimi, je veľmi pracná a je ťažké sa v nej vyznať a pochopiť ju do takej miery, aby bolo možné vedieť, ktoré informácie kde hľadať. Pre databázových architektov vznikajú zase iné otázky: Ako RDB model optimalizovať? Ktoré tabuľky a vzťahy sú kľúčové, alebo kritické? V súčasnosti sa tieto problémy riešia viacerými spôsobmi (databázové metriky a rankingy [1, 11-15], vizualizácia [17-27]) a existuje zopár nástrojov k takýmto účelom (IBM SPSS Modeler, Bellman data profiler, Polaris interactive DB visualization a iné).

2 Návrh riešenia

Mojím cieľom práce je vytvorenie postupu alebo metodiky na pochopenie a orientovanie sa v rozsiahlej alebo zložitej RDB. Na testovanie týchto postupov som používal kópiu databázy CEHZ, slúžiacej projektu "Centrálna evidencia hospodárskych zvierat SR". Databáza pracuje pod systémom Oracle a má veľkosť približne 2,4 GB. Neboli mi k nej poskytnuté žiadne ďalšie informácie a tak bude vhodne slúžiť na experimentálne účely. Postupy boli overené aj na ďalších dvoch reálnych databázach.

2.1 Získavanie metadát zo systémového katalógu

Niektoré metadáta o RDB je možné získať zo systémového katalógu. Sú to napríklad informácie o názvoch tabuliek a atribútov, rôzne obmedzenia a reštrikcie, primárne a cudzie kľúče, unikátne stĺpce, indexované stĺpce atď. K údajom sa dostaneme pomocou príkazov. Nie všetky tieto príkazy sú štandardizované (SQL) a preto si ich musíme naštudovať pre konkrétny RDB systém, v ktorom máme databázu uloženú.

O RDB CEHZ som zo systémových katalógov zistil, že obsahuje 332 tabuliek a 169 pohľadov, 2339 atribútov a medzi reláciami existuje 192 referenčných vzťahov. Z týchto parametrov môžeme teda považovať RDB CEHZ za pomerne zložitú a rozsiahlu. Ďalej je možné z týchto údajov vytvoriť ER diagram, ktorý by nám mal túto RDB umožniť pochopiť.

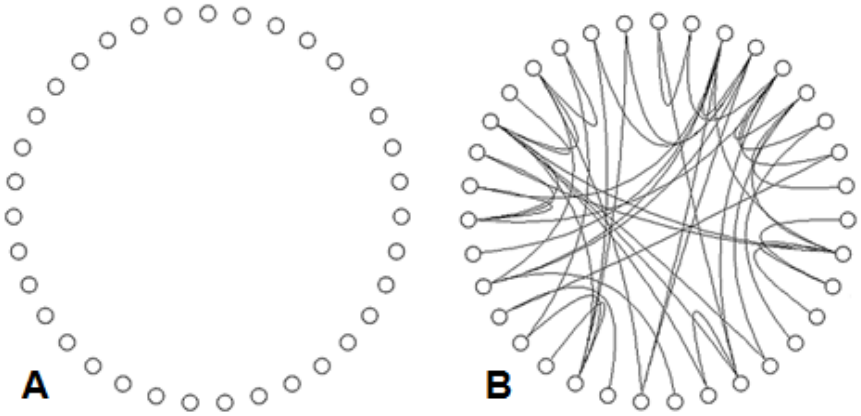
Tu narážame na prvý problém, ER diagram je príliš zložitý a ťažko sa v ňom orientuje. Keď v ňom chceme hľadať nejaké informácie, tak musíme pracne pozerať všetky tabuľky a sledovať vzťahy medzi nimi.

2.2 ER diagram pri rozsiahlych RDB

ER diagram je efektívna vizualizácia modelu RDB. Je to najpoužívanejšia pomôcka pre orientáciu sa v RDB. Pri rozsiahlych RDB vzhľadom na ich štruktúru, najmä počet vzťahov a tabuliek, je však ťažko použiteľná a zle sa v nej orientuje. Keď chceme používať ER diagram pri takýchto RDB, je dôležité aby boli tabuľky v ňom vhodne rozmiestnené. To by mal mať za úlohu autor, ktorý RDB schému navrhol.
Ďalším problémom ER diagramu je pre vizualizáciu rozsiahlych RDB ten, že nám neumožňuje rozoznať dôležité vzťahy a tabuľky. V ER diagrame vyzerajú všetky rovnocenne. Pri menšom počte tabuliek a vzťahov to nie je podstatné, keďže sme schopní si väčšinu z nich zapamätať. Pri väčšom počte tabuliek však potrebujeme odlíšiť tie podstatné a skúmať ich najskôr a až potom sa z nich dostať k menej dôležitým, podľa toho či ich potrebujeme skúmať alebo nie. Potrebujeme teda zobraziť v ER diagrame aj informácie o dôležitosti vzťahov a tabuliek.

2.2.1 Schemaball

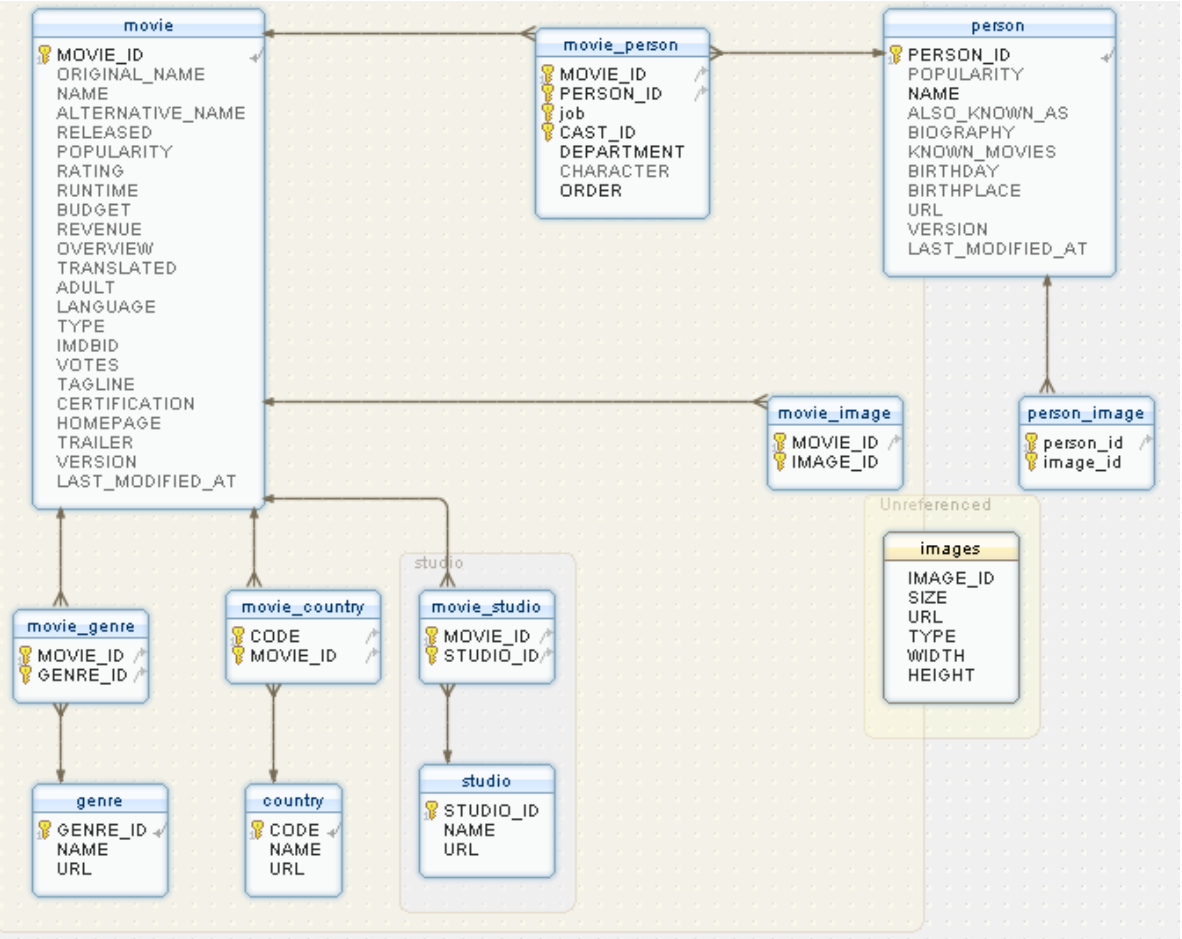
Myšlienkou zobrazovania schém rozsiahlych RDB sa už zaoberali v [22] a vymysleli vhodný typ vizualizácie. Nazvali ju Schemaball. Tento systém som prevzal, čiastočne upravil a vytvoril vlastnú softvérovú implementáciu (2.8 RDBAnalzyer). Schemaball zobrazuje schému RDB ako kružnicu kde tabuľky sú malé kruhy rovnomerne po nej rozmiestnené (viď obr. 1 A) a vzťah medzi dvoma tabuľkami je zobrazený ako bezierová krivka ktorá ich spája a jej vrchol smeruje ku stredu (viď obr. 1 B). Vedľa kružníc tabuliek smerom od stredu je možné vypisovať názvy tabuliek. V takejto forme by však vizualizácia slúžila horšie ako ER diagram.



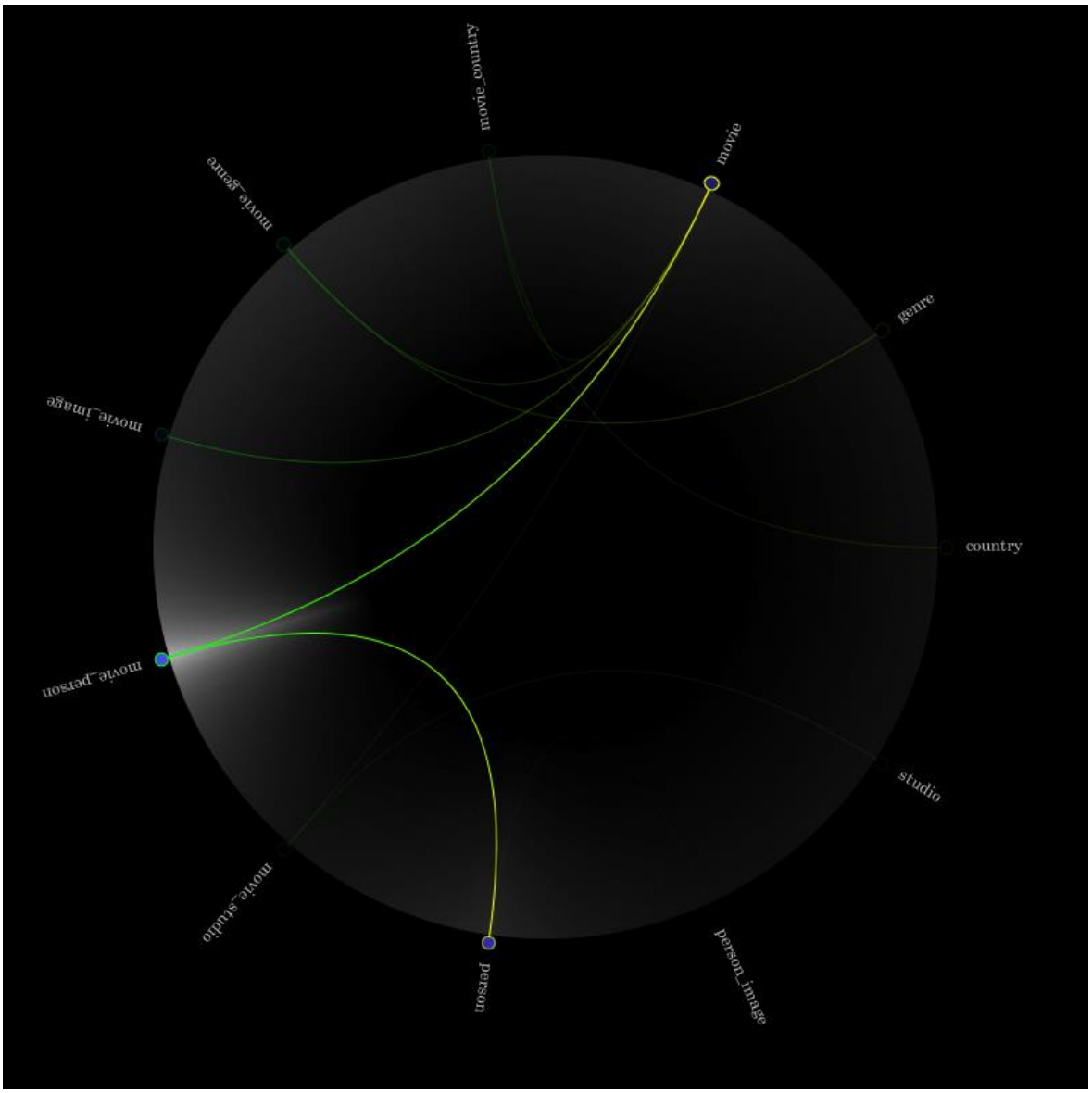
Obr. 1 *Malé kružnice reprezentujú tabuľky a bezierové krivky vzťahy medzi nimi.*

Pridanou hodnotou schemaballu je práve nedostatok ER diagramu a to zobrazenie dôležitosti vzťahov a tabuliek. Metriky nám slúžia na porovnávanie jednotlivých objektov v RDB. Tu práve využijeme metriky na ohodnotenie tabuliek a vzťahov. Tabuľky ohodnotíme podľa ich veľkosti metrikou TS (Table Size) [11]. Podľa [11] je to počet stĺpcov vynásobený počtom riadkov tabuľky. Podľa tejto hodnoty nastavíme jas farby kruhu reprezentujúceho tabuľku. Obdobným spôsobom zobrazíme aj vzťahy medzi tabuľkami pomocou metriky dôležitosť vzťahu (DV) [3]. Dôležitosť vzťahu vypočítame ako počet atribútov cudzieho kľúča vynásobený počtom riadkov, ktoré vzťah prepája tj. počet riadkov v tabuľke s cudzím kľúčom, kde je cudzí kľúč definovaný. Tieto metriky sú overené a často používané, je však na

nás, aby sme si ich ľubovoľne zmenili v prípade potreby. Pri vzťahoch je tiež dôležité zobraziť, ktorá tabuľka sa na ktorú odkazuje (vzťahy v RDB sú jednosmerné). Toto pôvodní autori Schemaball neriešili, upravil som teda zobrazovanie vzťahov tak, že krivka vzťahu nebude jednofarebná, ale bude lineárnym prechodom dvoch farieb počnúc jednou sýtou farbou na začiatku krivky končiac inou sýtou farbou na konci. Na obrázku 2 a 3 vidíme ER diagram a Schemaball rovnakej RDB. Ukážka je z menšej RDB, aby boli jasné a zobrazené všetky vzťahy. Pri rozsiahlej RDB nie je dôležité vidieť všetky vzťahy, ale hlavne vidieť tie podstatné. Tak isto, ak by mala tabuľka veľa menej dôležitejších vzťahov, tak by mala vyniknúť ako dôležitá, lebo veľa tabuliek spája. Hlavnou ideou Schemaballu a vizualizácie veľkého množstva dát nie je zobrazenie všetkého, ale hlavne dôležitého. Vykreslenie dát tak, aby sa v zobrazení dali hľadať skryté informácie, potláčali sa nepodstatné časti a zvýrazňovali sa tie podstatné. Sú dva spôsoby ako je to možné dosiahnuť. Jedna možnosť je extrahovať z dát dôležité časti pomocou metrík alebo agregáciou a potom tieto extrahované dáta zobrazovať. Inou možnosťou je vo vizualizácií vykresľovať všetky dáta a zabezpečiť, že vo výslednom zobrazení budú dôležité časti zvýraznené a menej dôležité nie. Schemaball som implementoval týmto druhým spôsobom pomocou metódy kreslenia alfa kanálu [2].



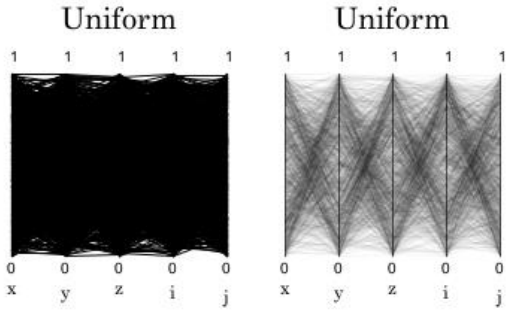
Obr. 2 *ER diagram RDB Filmy.*



Obr. 3 *Schemaball RDB Filmy. Podľa jasu farieb vidíme dôležité vzťahy a tabuľky. Vzťahy napr. z tabuľky movie_person začínajú zelenou farbou, čo znamená, že sa odkazuje na iné tabuľky.*

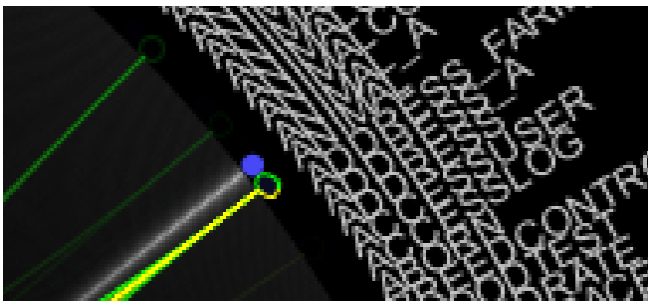
Táto metóda funguje tak, že pri vykresľovaní objektu sa okrem farby nastaví aj jeho priehľadnosť. Keď sa potom na už vykreslený objekt niečo nakreslí, výsledná farba bude závisieť od farieb a priehľadnosti oboch resp. všetkých prekrývajúcich sa objektov (viď obr. 4). V prípade našej vizualizácie je to požadované, lebo pri klasickom algoritme maliara[1] by sme nielen väčšinu vykresľovaných objektov nevideli kvôli tomu, že by boli prekreslené inými, ale výsledná vizualizácia by závisela aj od poradia vykresľovaných objektov, čo je absolútne nevhodné. V klasickom ER diagrame tento problém riešiť nemusíme, lebo tam sa nám žiadne objekty neprekrývajú a to je aj dôvod, prečo sa nám ich tam veľa nezmestí.

[1] Úplné prekreslenie podkladu vykresľovaným objektom. Paralela s maliarom, ktorý vždy pri ťahu štetcom prekryje pôvodný obraz.



Obr. 4 *Na obrázku vidíme tú istú vizualizáciu vľavo bez a vpravo s použitím alfa kanálu.*

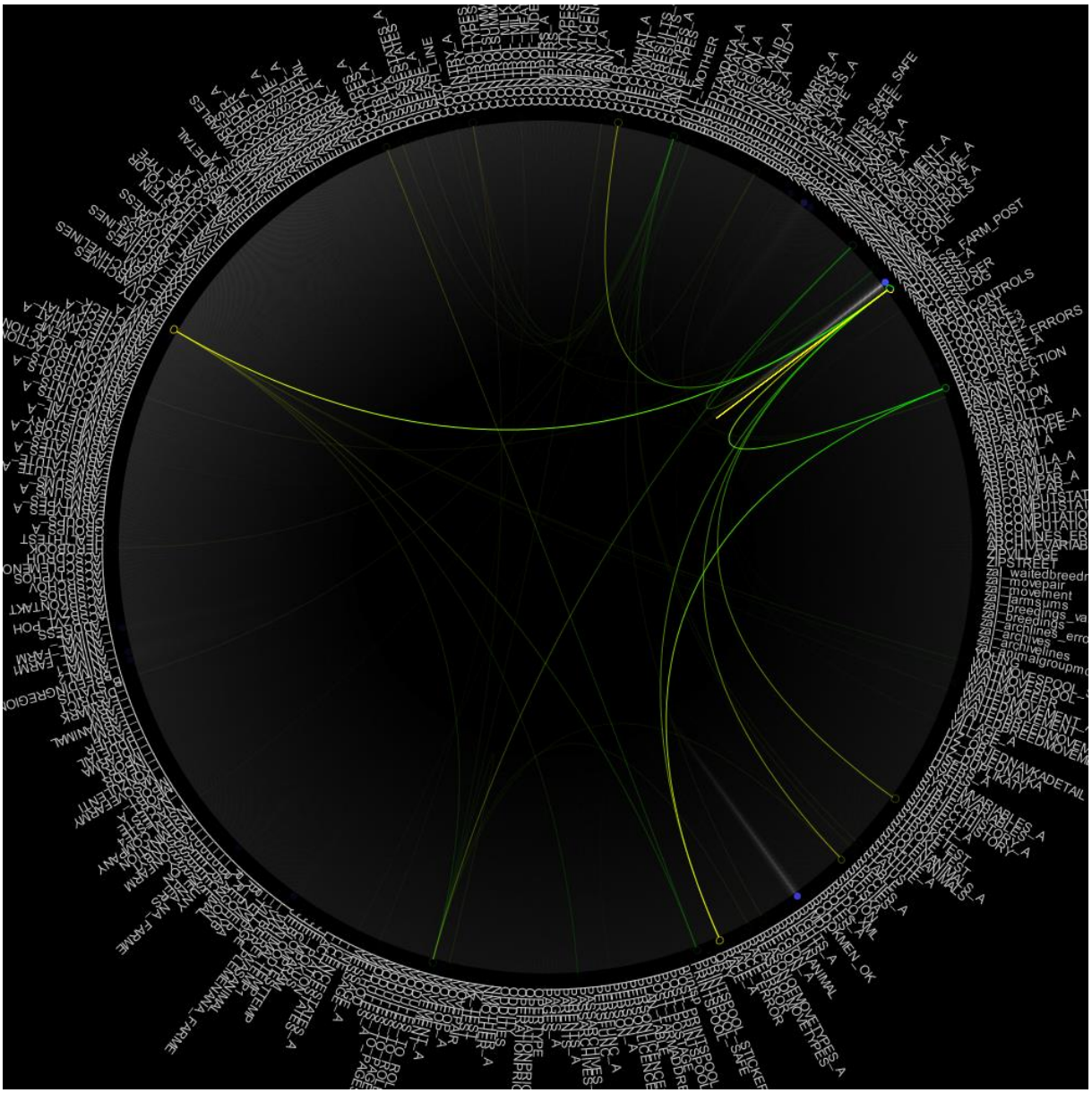
Na obrázku 6 vidíme vizualizáciu našej skúmanej RDB CEHZ pomocou Schemaball. Zvýraznené sú hlavné vzťahy a tabuľky, na ktoré sa máme v prvom rade pri skúmaní zamerať. Z vizualizácie vidíme, že väčšina kriviek intenzívnych farieb smeruje do tabuľky ANIMAL (viď obr. 5). Z tohto hľadiska je zrejme táto tabuľka najdôležitejšia, lebo je s jej riadkami prepojených veľa riadkov z iných tabuliek.



Obr. 5 *Detail vizualizácie pri tabuľke ANIMAL. Vzťahov s touto tabuľkou je buď veľa, alebo spája veľa riadkov, čo sa odráža na intenzite farby kriviek smerujúcich do nej. Farba kružnice tejto tabuľky nám reprezentuje pomer počtu riadkov v tabuľkách, ktoré sa odkazujú na riadky v tejto tabuľke (žltá) a počet riadkov v tabuľkách na ktoré sa odkazujú riadky tejto tabuľky (zelená). V prípade tabuľky ANIMAL je to približne 4 ku 6.*

Aj z názvu databázy CEHZ "Centrálna evidencia hospodárskych zvierat SR" je zrejmé, že hlavnou entitou bude "ANIMAL" (zviera). Tabuľka nad ňou ANIMAL_A je bez výraznejších vzťahov, ale intenzívna modrá farba indikuje, že je veľká. Pozrel som si bližšie stĺpce a pár záznamov týchto tabuliek a zistil som, že sa jedná o archív tabuľky ANIMAL, a že na riadky v tabuľke ANIMAL_A sa žiadne riadky z iných tabuliek neodkazujú a naopak. Takto je tu viacero tabuliek archivovaných s notáciou "_A" na konci. Tieto tabuľky by sme mali vylúčiť z ďalšieho skúmania, lebo nám žiadnu novú informáciu pre pochopenie RDB neposkytnú a ich vylúčenie nám zjednoduší model RDB. Ďalšie výrazné tabuľky buď prepojením alebo veľkosťou sú BLOODRACE, RACE, SPECIES, MOVEMENT, FARM, COLOR, CATTLE, BREEDING, ANIMALMOVEMENT, REGION a ďalšie. Len z tých názvov je možné sa domnievať, že RDB CEZH slúži na evidovanie a kategorizáciu zvierat, kde sa evidujú aj ich rodokmene a demografické údaje.

Ak nás zaujímajú ďalšie podrobnosti, môžeme si pozrieť konkrétne tabuľky a ich stĺpce kde zistíme, čo presne sa tam eviduje. Takýmto spôsobom môžeme preskúmať RDB od najdôležitejších tabuliek a vzťahov tak, aby sme vedeli, ak hľadáme v nej nejaké konkrétne informácie, jednak či ich môže obsahovať a kde by mohli byť. Taktiež nám tento postup umožní zistiť o akú RDB obsahovo ide.



Obr. 6 *Výsledná vizualizácia RDB CEHZ pomocou implementovanej Schemaball.*

Schemaball je vhodná vizualizačná metóda pre rozsiahle RDB s veľkým počtom tabuliek a vzťahov. Hlavné výhody oproti ER diagramu sú:

- zvýrazňovanie dôležitých tabuliek a vzťahov,
- do relatívne malého obrázka zobrazíme celú RDB,
- zobrazuje nielen schému RDB, ale aj je obsah. Tento bod je podľa mňa najdôležitejší, lebo ak chceme pochopiť nejakú RDB, záleží najmä na dátach v nej o akú databázu ide,
- Schemaball je tiež možné využiť na zobrazenie hocijakých hranovo a vrcholovo ohodnotených grafov.

Hlavnou nevýhodou Schemaball je, že nedokážeme na základe nej priamo, tak ako s ER diagramom, vytvárať SQL dotazy. Pre vytváranie dotazov by musela byť interaktívna s možnosťou dodatočného zobrazenia atribútov.

2.3 Distribučná funkcia veľkosti tabuliek (TS) a dôležitosti vzťahov (DV)

Z obrázka 6 je nám jasné, že pri reálnej RDB nie sú hodnoty veľkosti tabuliek a tiež ani dôležitosti vzťahov rozložené rovnomerne. To som tiež predpokladal, ale nečakal som, že budú medzi tými hodnotami až také skoky. Urobil som teda štatistické testy na určenie z akého rozdelenia pravdepodobnosti by mohli pochádzať tieto hodnoty. Skúmal som konkrétne veľkosti tabuliek podľa metriky TS (počet stĺpcov krát počet riadkov), dôležitosť vzťahov podľa metriky DV (počet atribútov cudzieho kľúča krát počet riadkov, ktoré ho majú definovaný). Oba súbory hodnôt som štatisticky spracoval v programe Input Analyzer, ktorý je súčasťou programu Arena od firmy Rockwell Software. Oba štatistické súbory pochádzajú z Weibulového rozdelenia na hladine významnosti 95 % (viď tab. 1). Softvér robí 2 štatistické testy na určenie vhodného rozdelenie pravdepodobnosti a jeho parametrov (chí kvadrát test dobrej zhody a Kolmogorov-Smirnov test, oba musia potvrdiť hypotézu na určenej hladine významnosti, v tomto prípade 95%).

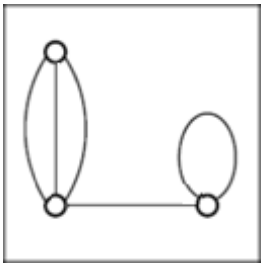
Štatistický súbor	**Zistené rozdelenie pravdepodobnosti**	**Priemerná štvorcová chyba**
veľkosť tabuliek (TS)	WEIB(5.66e+003, 0,234)	0,001896
dôležitosť vzťahov (DV)	WEIB(1,22e+004, 0,225)	0,000378

Tab. 1 *Zistené rozdelenia pravdepodobnosti štatistických súborov aj s priemernou štvorcovou chybou na hladine významnosti 95%.*

Skúmal som distribučnú funkciu aj na ďalších dvoch reálnych databázach, kde bolo rozdelenie rovnaké s menšou zmenou koeficientov. Vyplýva nám z toho, že aj keď je tabuliek a vzťahov v databáze veľa, dôležitých bude pri Weibulovom rozdelení vždy len nepatrná časť, ktorú je potrebné skúmať. Takto sa aj v zložitých relačných modeloch dá rýchlo orientovať.

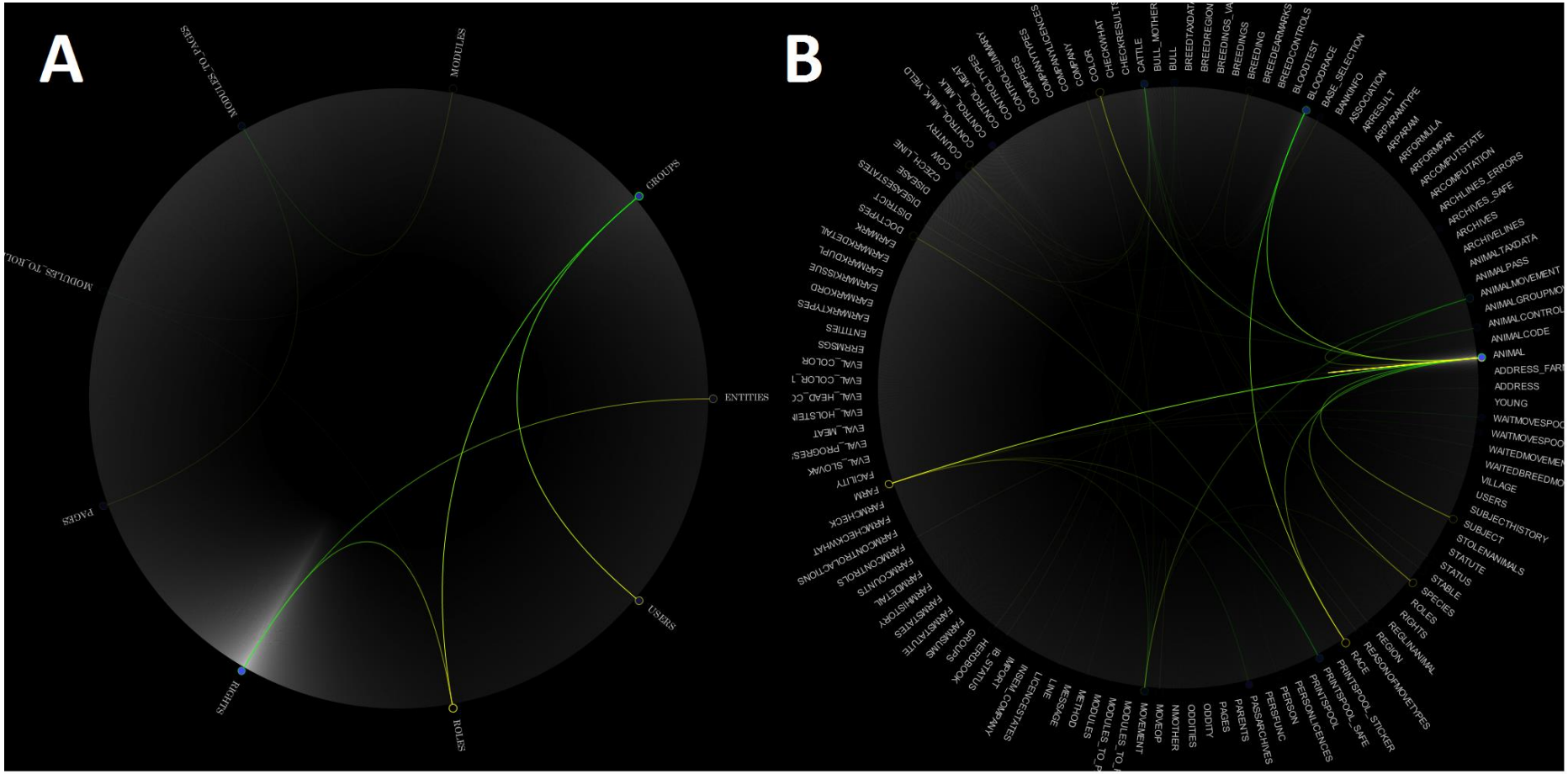
2.4 Rozdelenie grafu schémy RDB na komponenty

Ako bolo spomínané v predchádzajúcej kapitole, pri zobrazení RDB CEHZ sme našli tabuľky, ktoré slúžili na archiváciu iných tabuliek. Neboli prepojené vzťahmi so žiadnymi inými tabuľkami. Takéto tabuľky, ktoré nie sú prepojené s inými, nám zbytočne zneprehľadňujú vizualizáciu a pôsobia mätúco. Je vhodné takéto tabuľky odstrániť. Samozrejme, že pred tým ako ich odstránime, ich analyzujeme, aby sme zistili, aký majú význam v RDB. Tento princíp oddelenia samostatných častí RDB sa dá zovšeobecniť tak, že rozdelíme celú schému RDB na samostatné komponenty. Schému RDB si najskôr prevedieme do grafovej formy (viď obr. 7). Vrcholy grafu budú tabuľky a neorientované hrany medzi nimi budú vzťahy medzi tabuľkami.



Obr. 7 *Grafová reprezentácia RDB schémy. Tabuľky sú vrcholy grafu a neorientované hrany sú vzťahy medzi nimi. Takýto graf pri RDB sa nazýva pseudograf.*

Na takto vytvorený graf použijeme napríklad Tarryho algoritmus, ktorý nám rozdelí graf na jednotlivé komponenty. Tieto komponenty následne skúmame ako samostatné celky. Ideálnym prípadom je, ak zistíme, že RDB sa skladá z viacerých menších nezávislých databáz. Tie sa dajú následne jednoduchšie pochopiť. Po aplikovaní postupu na skúmanú RDB CEHZ sme dostali dva viactabuľkové komponenty a zvyšok boli samostatné tabuľky. Po preskúmaní samostatných tabuliek som zistil, že všetky sú archívne, keďže boli nazvané ako tabuľka, ktorú archivujú s príponou "_A" a obsahovali rovnaké stĺpce. Zvyšné dva komponenty som si znova zobrazil pomocou vizualizácie Schemaball (viď obr. 8). Prvý, menší komponent (na obrázku 8 – A, 9 tabuliek) je databáza webového rozhrania ku prístupu k hlavnej databáze (druhého komponentu) rozdelená do modulov (tabuľky PAGES, MODULES_TO_PAGES, MODULES). Je tu uložený tiež systém práv užívateľov (tabuľky USERS, GROUP, ROLES, RIGHTS). Autori RDB CEHZ zrejme chceli mať všetko pokope v jednej RDB, ale bez ich dodatočnej informácie alebo rozdelenia grafu schémy na komponenty, by sme na to len sotva prišli. Druhý, najväčší komponent (118 tabuliek) je vlastne hlavná databáza, ktorá je teraz už očistená od priamo s ňou nesúvisiacich častí, ktoré nám ju komplikovali. Je však stále dosť veľká na to, aby sme sa v nej orientovali pomocou ER diagramu. Na obrázku 8 - B je možné identifikovať hlavné tabuľky a vzťahy (ANIMAL, RACE, FARM, BLOODRACE, SPECIES, SUBJECT, MOVEMENT, CATTLE, COLOR a ďalšie). Skúmaním týchto tabuliek a dát v nich som zistil, že RDB CEHZ je zameraná na evidenciu hospodárskych zvierat (hovädzí dobytok, ošípané, ovce, kozy, hydina a pštrosy). Evidujú sa tu farmy zo slovenských regiónov, rodokmene zvierat, ich choroby, očkovania, kontrola mäsa z týchto zvierat. Takýmto spôsobom je možné pochopiť veľkú RDB, orientovať sa v nej a získavať z nej informácie.



Obr. 8 *Výsledné dva nezávislé komponenty skúmanej RDB CEHZ.*

2.5 Navrhnutá metóda pre orientáciu a pochopenie rozsiahlej RDB

Na základe použitia databázových metrík, vizualizácie a grafového algoritmu z predchádzajúcich podkapitol bola vytvorená metóda, ktorá umožňuje efektívny spôsob orientácie sa a pochopenia rozsiahlej RDB. Je vhodná na analýzu neznámej RDB, ktorú chceme pochopiť, ale taktiež aj na známu RDB, ktorá je príliš veľká na to, aby sme sa z pamäti vedeli v nej orientovať. Vytvorená metóda pozostáva z nasledovných krokov:

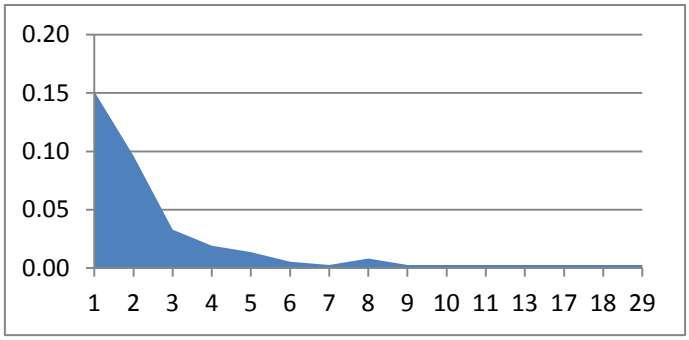
1. Rozdeľ neorientovaný graf RDB schémy (graf tvoria tabuľky a vzťahy cudzích kľúčov medzi nimi) na komponenty napríklad pomocou Tarryho algoritmu,
2. Skúmaj každý komponent grafu (krok 3) osobitne ako samostatnú RDB od komponentu s najmenším počtom tabuliek.
3. Ak je tabuľka samostatná (bez vzťahov), je dosť pravdepodobné, že bude archívna. To sa dá overiť jednoducho. Treba sa pozrieť na ostatné tabuľky aj z iných komponentov, či nie sú podobné názvami. Ak sú, tak pozrieme ešte názvy stĺpcov, ktoré by sa mali zhodovať (nemusia všetky). Na základe riadkov v tabuľke určíme spôsob archivácie (za aké časové obdobie sa robí a pod.). Ak nenájdeme žiadnu podobnú tabuľku a zistíme teda, že nejde o archívnu tabuľku, malo by byť možné len na základe názvov stĺpcov a dát v nej určiť jej význam.
   Ak komponent tvorí viac ako jedna tabuľka, na základe veľkosti komponentu (počet tabuliek) určíme, či ide o hlavnú časť RDB, alebo nejaké podčasti priamo nezávislé na zvyšku RDB. Je nepravdepodobné, že RDB obsahuje viac samostatných RDB, kde všetky sú rovnocenné a nezávislé lebo tvorca RDB by ich vytvoril v osobitných schémach. Ak sa nachádzajú v jednej schéme viaceré RDB, bude to len z toho dôvodu, že nejako, nie priamo, so sebou súvisia. (Napríklad tak ako v našom praktickom príklade, kde menší komponent tvoril RDB pre webové rozhranie na prístup, zmenu dát a systém práv hlavného komponentu.)
   Následne si komponent zobrazíme pomocou ER diagramu, kde sa na základe názvov tabuliek, ich stĺpcov, dát v nich a vzťahoch budeme snažiť pochopiť databázu. Ak by bol ER diagram príliš veľký na pochopenie, použijeme vizualizáciu Schemaball a budeme skúmať podstatné tabuľky a vzťahy, ktoré budú výrazne označené. Na základe nich zistíme zhruba aké informácie RDB uchováva a čo je jej účelom. Pre konkrétnejšie detaily budeme z týchto kľúčových tabuliek prechádzať k menej dôležitejším cez vzťahy, vždy podľa toho, čo chceme zistiť. Popri vizualizácií Schemaball je nutné mať k dispozícií nejaký nástroj, ktorý by nám zobrazoval stĺpce a dáta tabuliek, lebo Schemaball tieto informácie kvôli prehľadnosti nezobrazuje.

Tento postup sa dá využiť aj pri iných ako relačných databázach (objektové, sieťové), vyžadovalo by to však menšie zmeny.

2.6 Bezškálové charakteristiky rozsiahlych RDB

Pri skúmaní schémy RDB CEHZ som si všimol pri použití metriky NR (Number of References – počet vzťahov tabuľky), že väčšina tabuliek zo schémy má len málo vzťahov, ale existuje zopár tabuliek, ktoré ich majú výrazne veľa. Graf na obrázku 9 nám zobrazuje distribúciu počtu vzťahov tabuliek (inými slovami, aká je pravdepodobnosť, že má tabuľka x - vzťahov). Z grafu vidíme, že drvivá väčšina tabuliek nemá viac ako 3 vzťahy. Naopak, existujú aj tabuľky, ktoré majú 17, 18 a aj 29 vzťahov. Táto distribučná funkcia sa nápadne podobá na distribučnú funkciu stupňov vrcholov v bezškálových sieťach. Je dosť pravdepodobné, že aj graf RDB schémy[2] je bezškálová sieť.

[2] Neorientovaný graf kde tabuľky sú vrcholy a vzťahy medzi nimi sú hranami grafu.

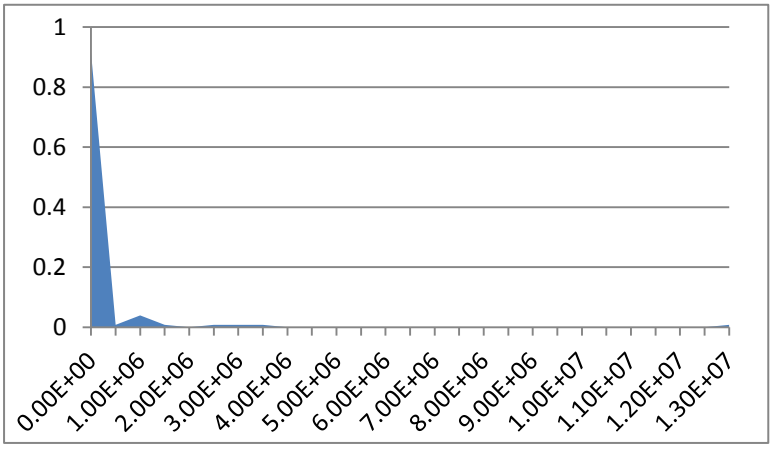


Obr. 9 *Distribučná funkcia počtu vzťahov tabuliek. Na x-ovej osi je počet vzťahov a na y-lonovej osi príslušný podiel tabuliek s takýmto počtom vzťahov.*

Bezškálové siete vznikajú rozširovaním existujúcej siete o nové uzly a ich preferenčným pripojovaním [28]. Rozsiahle RDB vznikajú tiež rozširovaním existujúcich RDB a pridávaním tabuliek. Či sa pripájajú preferenčne, už nie je také jednoznačné. Ďalším znakom bezškálových sietí je existencia centier [28]. V RDB CEHZ existujú. Ak by sa ukázalo, že graf rozsiahlych RDB je bezškálová sieť, mohli by sme jej vlastnosti využiť pri skúmaní RDB. Na zistenie či je sieť bezškálová sa väčšinou skúma len distribučná funkcia stupňov vrcholov grafu. Albert Lászlo Barabási definoval bezškálové siete ako siete s mocninovým chvostom distribučnej funkcie stupňov vrcholov grafu (orig. "Networks with a power law tail in their degree distribution are called scale-free networks") [28]. Na obrázku 9 vidíme mocninovú funkciu aj s takzvaným ťažkým chvostom. Bezškálové siete sú väčšinou rozsiahle, rádovo státisíce, milióny uzlov. Avšak aj túto sieť môžeme považovať za bezškálovú i keď je malá. Graf je možné vytvoriť aj iným spôsobom ako zo schémy RDB. Môžeme zakomponovať do neho aj obsah RDB. Navrhol a otestoval som teda ďalšie dva prístupy.

2.6.1 Prístup 2

Tabuľky budú uzly a hrana medzi dvoma uzlami bude za každý záznam prepojený medzi tabuľkami. Napríklad ak má tabuľka A - 10 záznamov, ktoré sa odkazujú na záznamy v tabuľke B, tak medzi tabuľkami A a B je 10 hrán. Tento prístup rozdelí poradie centier vzhľadom na počet záznamov a ich prepojenia s ostatnými záznamami v iných tabuľkách. Graf distribučnej funkcie takejto sieti je na obrázku 10. Vidíme, že uzly majú hrán rádovo v miliónoch a centrá sa oddelili ešte výraznejšie.

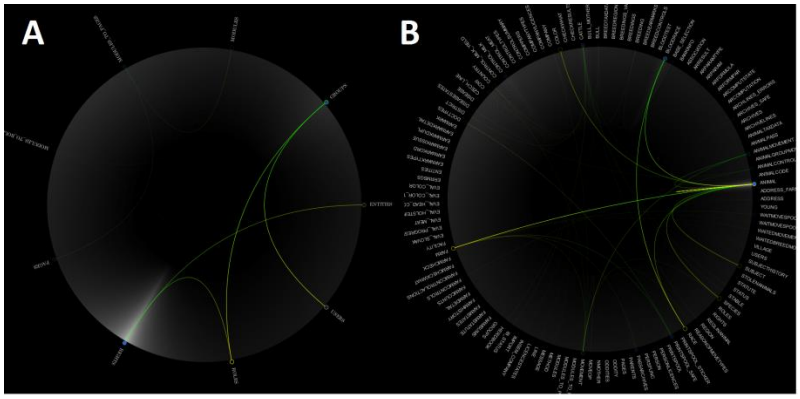


Obr. 10 *Distribučná funkcia počtu vzťahov tabuliek (vzťah medzi tabuľkami je za každý prepojený záznam). Na x-ovej osi je počet vzťahov a na y-lonovej osi príslušný podiel tabuliek s takýmto počtom vzťahov. V okolí hodnôt 3e+06 a 1,3e+07 na x-ovej osi sú zaznačené centrá.*

Výhodou prístupu je, že sieť je závislá najmä na dátach v RDB. Čiže centrá nám vzniknú tam, kde je najviac prepojených záznamov. Málo prepojené tabuľky, čo majú veľa vzťahov by v tomto prípade v sieti neboli dôležité, čo je žiadúce.

2.6.2 Prístup 3

Vytvorenie grafu môžeme ešte vylepšiť tak, že graf nám budú tvoriť samotné dáta v RDB. Čiže vrcholy grafu budú riadky (n-tice) z tabuliek a hrany budú referencie medzi konkrétnymi riadkami. Napríklad ak je v tabuľke OSOBA evidovaný záznam pod unikátnym ID=11, tento riadok bude vrcholom grafu a hrany bude mať iba ak sa nejaký záznam z ľubovoľnej tabuľky odkazuje na ID=11 alebo ak sa záznam s ID=11 odkazuje na iné riadky v ľubovoľných tabuľkách. Vytvoriť však takýto graf z rozsiahlej RDB nie je triviálna úloha. Vytvoril som na to program, ktorý túto transformáciu urobí pre všetky RDBS s podporou JDBC . Generovanie grafu pre RDB CEHZ trvalo niečo vyše dňa. Vytvorený graf mal okolo 30 miliónov vrcholov a 11 miliónov hrán. Veľký počet vrcholov nemal žiadnu hranu cca 25 miliónov. Tieto vrcholy nás nezaujímajú, keďže nie sú prepojené. Boli najmä z archívnych tabuliek. V kapitole 2.4 (Rozdelenie grafu schémy RDB na komponenty) sa nám RDB CEHZ rozdelila na dva komponenty (komponenty A, B obr. 11, detail, obr. 8 na str. 11).



Obr. 11 *Dva Komponenty grafu schémy RDB CEHZ.B je hlavná databáza a A je časť, ktorá zabezpečuje webové rozhranie a systém práv prístupu. (detail obrázka je na str. 11, obr. 8)*

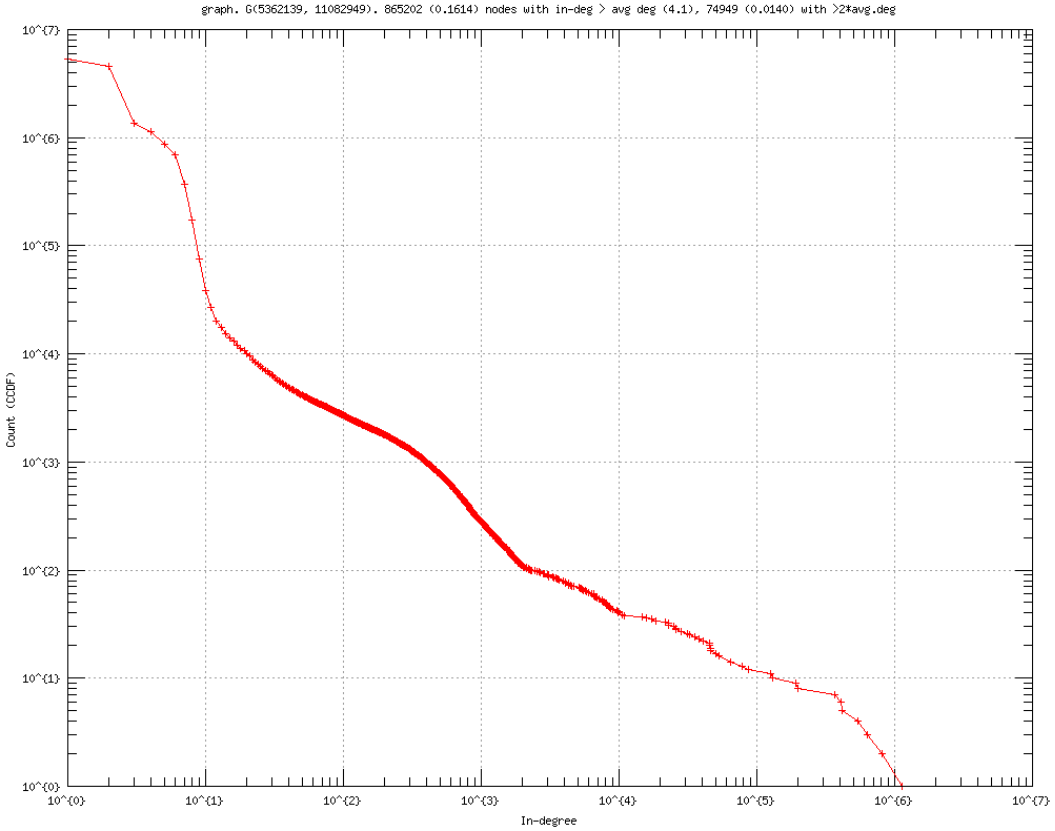
Tieto komponenty som skúmal osobitne, keďže ak ich grafy schém sú osobitné komponenty, tak určite nebudú prepojené ani dátami. Hlavný komponent B (obsahuje takmer všetky dáta) bol na moje prekvapenie súvislý. Tj. neexistuje žiaden riadok tabuľky resp. záznam, ktorý by nebol súčasťou jednej veľkej prepojenej sieti záznamov. Z druhého komponentu A (táto časť RDB zabezpečuje webové rozhranie a systém práv) vznikol graf so 7mimi komponentmi. Rozdelenie vzniknutých komponentov z oboch databáz je v tabuľke 2.

Komponent	Počet vrcholov	Databáza	Zoznam Tabuliek z ktorých sú záznamy
1	5 359 342	HLAVNÁ - B	Všetky (118)
2	2 755	A	Všetky (9)
3	17	A	ENTITIES, MODULES, MODULES_TO_PAGES, MODULES_TO_ROLES, PAGES, RIGHTS, ROLES
4	16	A	MODULES, MODULES_TO_PAGES, PAGES
5	3	A	MODULES, MODULES_TO_PAGES, PAGES
6	2	A	MODULES_TO_PAGES, PAGES
7	2	A	MODULES_TO_PAGES, PAGES
8	2	A	MODULES_TO_PAGES, PAGES

Tab. 2 *Vzniknuté komponenty z grafov záznamov. (Na spracovanie grafu bola použitá knižnica SNAP [29])*

Aj pri druhej databáze A sú však takmer všetky dáta (98%) v jednom komponente. Nie je to zrejme náhoda, že graf záznamov v databáze je tak prepojený, že tvorí jednu sieť a len málo

záznamov k nej nie je pripojených. To je tiež jedna z charakteristík bezškálových sietí. Distribúcia stupňov vrcholov grafu nám to potvrdzuje (Obr. 12).



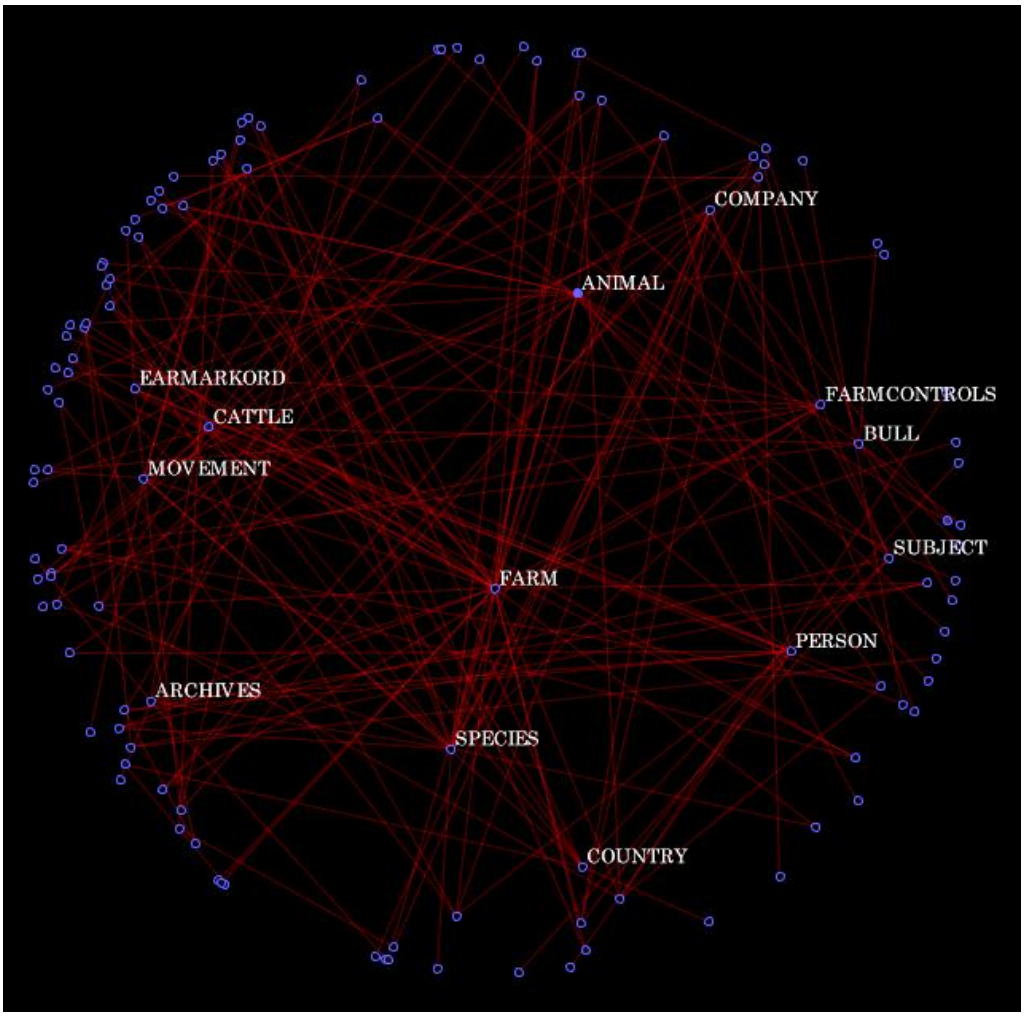
Obr. 12 *Distribučná funkcia stupňa vrcholov grafu Na x-ovej osi je počet vzťahov a na y-lonovej osi príslušný počet vrcholov s takýmto počtom vzťahov. (Graf je vygenerovaný pomocou knižnice SNAP [29], osi rastú exponenciálne.)*

Vo všetkých troch prístupoch nám vychádza graf vytvorený z RDB ako bezškálová sieť. V bezškálových sieťach sú centrá najdôležitejšími uzlami siete, ktoré držia celú sieť pohromade. V grafoch vytvorených z RDB sú to tabuľky alebo záznamy (prístup 3), ktoré sú v schéme kľúčové. Tak ako pri bezškálových sieťach, ak chceme sieť dobre pochopiť, tak musíme skúmať jej centrá. Pri RDB teda tiež v prvom rade skúmame tabuľky resp. záznamy, ktoré sú centrami. Tiež sa treba zamerať aj na ostatné vrcholy a hrany grafu, po ktorých vylúčení sa nám rozdelí graf na komponenty. Tieto vrcholy, hrany a centrá musíme tiež dobre zabezpečiť, lebo ich odstránenie nám rozdelí celú RDB na komponenty a väčšina z informácií, ktoré v nej ostanú bude nepoužiteľná lebo ich nebudeme vedieť spolu prepojiť.
Pri pohľade na RDB ako na sieť nám teda dôležitosť tabuliek a záznamov určuje počet ich vzťahov. Sú to vlastne databázové metriky. Prvým prístupom sme overili spôsobilosť jednej z existujúcich metrík a vďaka ďalším dvom prístupom môžeme zaviesť dve nové metriky:

- Metrika 1 - Počet vzťahov tabuliek s inými tabuľkami, kde vzťah medzi tabuľkami je ak tabuľka A obsahuje cudzí kľúč, ktorý sa odkazuje na stĺpec/e v tabuľke B. Nie je dôležitá orientácia vzťahov, vzťahy považujeme za neorientované. (overili sme použiteľnosť už existujúcej metriky - Number of References, NR (T))
- Metrika 2 - Druhá metrika zohľadňuje aj obsah RDB. Tabuľka A má taký počet vzťahov s tabuľkou B, koľko existuje záznamov v tabuľke A, ktoré sa odkazujú cez cudzí kľúč na záznamy v tabuľke B a naopak. Čiže ešte pripočítame k nim aj počet záznamov v tabuľke B, ktoré sa cez cudzí kľúč odkazujú na tabuľku A. Vzťahy sú neorientované a pre tabuľku je to suma vzťahov so všetkými ostatnými tabuľkami.

- Metrika 3 – Počet vzťahov záznamu (n-tice). Táto metrika nie je tabuľková, ale riadková.

Na jednoduché identifikovanie centier a ich dôležitosti som vytvoril vizualizáciu (viď obr. 13). Modré kruhy reprezentujú tabuľky. Intenzita ich vnútorného zafarbenia identifikuje ich veľkosť na základe metriky (TS – definovaná v 2.2.1). Tabuľky sú sústredené do kruhu, rozmiestnené sú náhodne po obvode kružnici, pričom vzdialenosť každej tabuľky od stredu je nepriamoúmerná jej počtu vzťahov, vypočítaného podľa metriky 1 (čím viac vzťahov má tabuľka tým je bližšie pri strede). Intenzita farby vzťahu identifikuje dôležitosť vzťahu na základe metriky (DV – definovaná v 2.2.1). Z vizualizácie je vidieť, že väčšina tabuliek je rozmiestnená na okraji kruhu (väčšina tabuliek je len pripojená k nejakému centru) a pár tabuliek - centier je blízko stredu. Takouto vizualizáciou sa dajú zobrazovať aj grafy sociálnych sieti resp. všetky bezškálové siete.

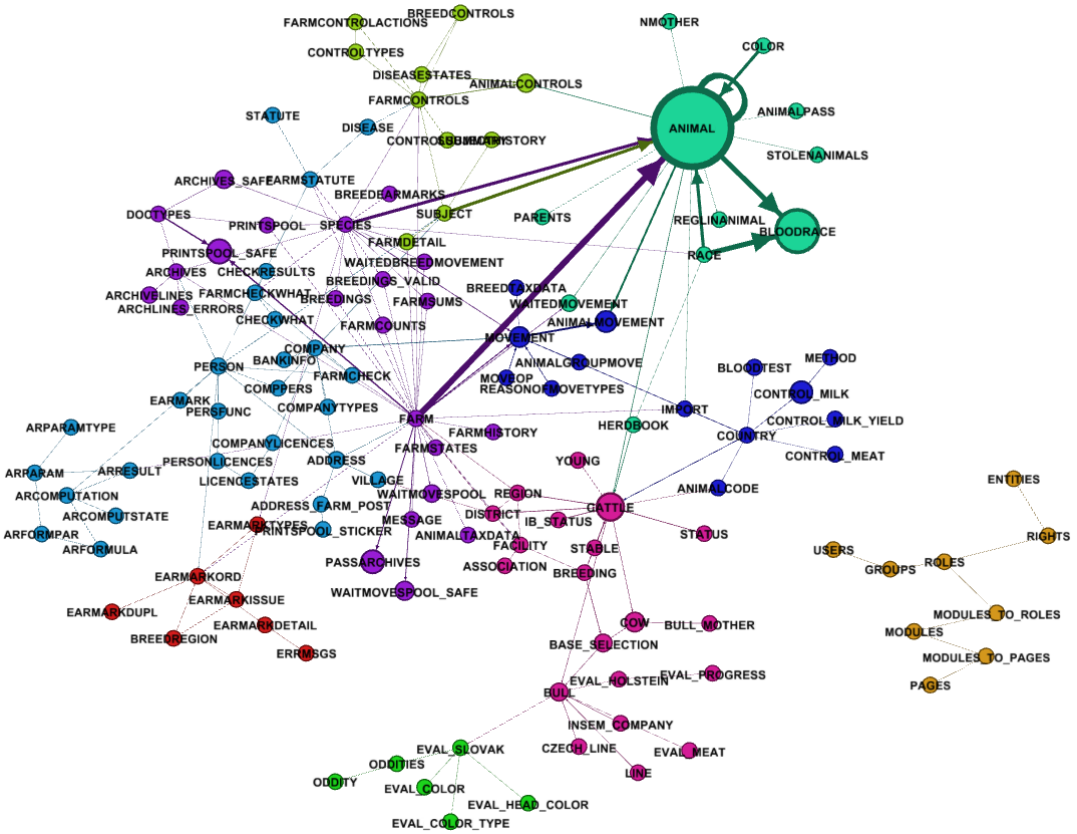


Obr.13 *Vizualizácia vzťahov a tabuliek v RDB CEHZ, modré kruhy reprezentujú tabuľky. Čím sú bližšie ku stredu, tým majú viacej vzťahov. Vizualizácia nám jasne oddelí centrá od ostatných tabuliek.*

2.7 Skúmanie a vizualizácia grafu RDB

Transformovať RDB do grafu sa dá viacerými spôsobmi (len v predchádzajúcej kapitole som spomenul 3 prístupy). Zoberme si ten najjednoduchší spôsob, vrcholy sú tabuľky a orientované hrany sú vzťahy cudzích kľúčov medzi nimi. Takýto graf vlastne zobrazuje aj ER diagram. Konkrétne pre RDB CEHZ som ho vizualizoval pomocou nástroja na

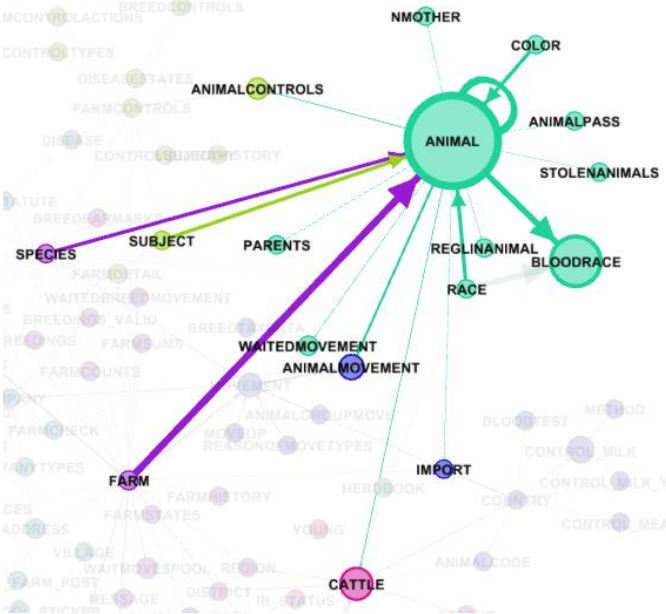
vizualizovanie grafov Gephi [36] . Nástroj ponúka viacero algoritmov pre automatické rozmiestnenie vrcholov grafu vo vizualizácii. Použitím algoritmu Force Atlas som dostal najprehľadnejší výsledok. Použil som aj automatickú klasterizáciu vrcholov na základe ich vzťahov. Na obrázku sú farebne odlíšené jednotlivé klastre (triedy). Vizualizácia si vyžadovala ešte menšiu interaktívnu úpravu rozmiestnenia, aby boli názvy tabuliek ako tak prehľadné. Ako môžeme vidieť na obrázku 14, samotný algoritmus nám rozdelil dva komponenty grafu a umiestnil vrcholy s veľkými stupňami (centrá) do stredu klastrov. Pri spoznávaní a orientovaní sa v schéme RDB opäť postupujeme od centier. Vizualizáciu môžeme vylepšiť pridaním metrík do grafu. Ohodnotenie hrán podľa metriky DV (dôležitosť vzťahov), a vrcholov podľa metriky TS (veľkosť tabuliek) definovaných v kapitole 2.2.1 nám pridá do vizualizácie ďalšie informácie. Vo vizualizácii to bude znamenať väčšie vrcholy a hrany s vyšším ohodnotením. Na obrázku 14 je graf schémy CEHZ aj s príslušnými metrikami. Táto vizualizácia si vyžadovala pomerne veľa úprav. Z kapitoly 2.3 vieme, že veľkosti tabuliek a dôležitosti vzťahov nerastú v schéme lineárne, ale exponenciálne. Vo vizualizácií musíme preto nastaviť, aby veľkosť hrany alebo vrcholu nebola od ohodnotenia závislá lineárne, ale napríklad logaritmicky. V prípade implicitného nastavenia nástroja (lineárna veľkosť v závislosti od ohodnotenia) by sme videli dva vrcholy a pár vzťahov, ktoré by všetko ostatné prekryli. Ďalej bolo nutné ešte čiastočne upraviť rozmiestnenie, aby bola čitateľná väčšina názvov tabuliek. Algoritmus rozmiestnenia Force Atlas bral do úvahy aj ohodnotenia hrán a vrcholov.



Obr. 14 *Vizualizácia grafu schémy CEHZ, pridanie ohodnotenia hrán a vrcholov na základe čoho je zobrazená hrúbka hrany resp. veľkosť vrcholu. Vizualizácia si vyžadovala aj ručnú úpravu rozmiestnenia vrcholov kvôli prehľadnosti.*

V takejto vizualizácii, kde sú hrany a vrcholy ohodnotené, sa môžeme prirodzene zamerať pri skúmaní a orientovaní sa v RDB najskôr na podstatné tabuľky a vzťahy, čo nám urýchli proces získavania dôležitých informácii o RDB. Ďalšou výhodou je, že graf nám zobrazuje

nepriamo aj informácie o dátach, čo je dôležité. Vizualizačný nástroj Gephi umožňuje po nadídení kurzoru myši nad vrchol zobrazenie všetkých priamo prepojených vrcholov a hrán k nemu. Orientácia v grafe sa stáva tak ešte prehľadnejšia. Na obrázku 15 je ukážka priamych vzťahov s vrcholom (tabuľkou) ANIMAL. Vizualizačný nástroj Gephi nám ponúka veľmi dobré riešenie orientácie sa vo veľkej RDB schéme. Interaktívna vizualizácia nám umožňuje jednoduché skúmanie RDB. V prípade dodatočných informácii k tabuľkám (napr. názvy stĺpcov, primárne a cudzie kľúče, atď.) môžeme dokonca vytvárať SQL príkazy. Nedostatky riešenia sú najmä v tom, že algoritmy na rozmiestnenie grafu vo vizualizácií nie sú úplne dokonalé a prehľadné a rozmiestnenie si vyžaduje dodatočné interaktívne úpravy. Ďalším nedostatkom je, že pri aplikovaní metrík na veľkosť hrán a vrcholov je potrebné tiež dodatočné ručné vyladenie vizualizácie, aby sme ju mali prehľadnú a použiteľnú. Export z ľubovoľnej RDB aj s metrikami do Gephi som implementoval v nástroji RDBAnalyzer (2.8 Vytvorenie nástroja RDBAnalyzer).



Obr. 15 *Interaktívne zobrazenie tabuľky ANIMAL a jej priamych vzťahov.*

2.8 Vytvorenie nástroja RDBAnalyzer

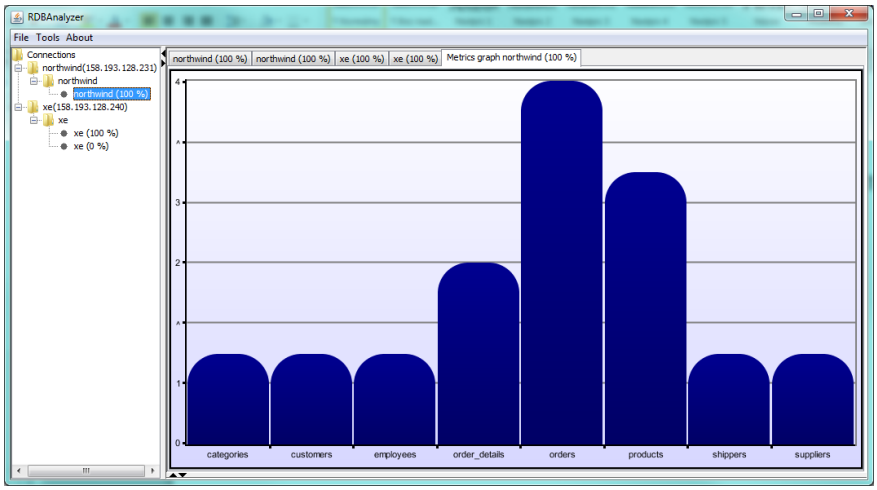
Metódy a postupy z predošlých kapitol som implementoval a združil do softvérového nástroja RDBAnalyzer. Nástroj má slúžiť ako pomôcka pre ľudí pracujúcich s veľkými RDB, v ktorých sa chcú rýchlejšie orientovať a pochopiť ich, na rýchle zorientovanie sa v RDB o ktorých nemáme žiadne informácie a chceme zistiť načo slúžia, aké informácie obsahujú, na údržbu rozsiahlych RDB a odhalenie kľúčových tabuliek, vzťahov a dátových tokov.

Aktuálne sú implementované v nástroji RDBAnalyzer nasledovné funkcionality:

- rozdelenie grafu schémy na komponenty,
- vizualizovanie pomocou Schemaball,
- vizualizovanie grafu schémy a nájdenie centier,
- zobrazenie stĺpcového grafu tabuliek a vzťahov podľa rôznych metrík,
- export grafu RDB aj metrikami do súboru “gexf“ pre spracovanie v nástroji Gephi [36].

Nástroj RDBAnalyzer odporúčam používať spolu s proprietárnymi nástrojmi na manažovanie RDB. V nich sa dá zistiť veľa dodatočných informácií o RDB a obsahu tabuliek. Navyše niektoré majú možnosti vizualizovania ER diagramov, modelovania schémy a RDB analýz. RDBAnalyzer poskytuje funkcie, ktoré štandardné proprietárne nástroje neposkytujú avšak

neposkytuje všetky štandardné funkcie týchto nástrojov. Je modulovo navrhnutý, ľahko rozšíriteľný a podporuje všetky RDBS podporujúce JDBC. Ukážku nástroja môžeme vidieť na obrázku 16.



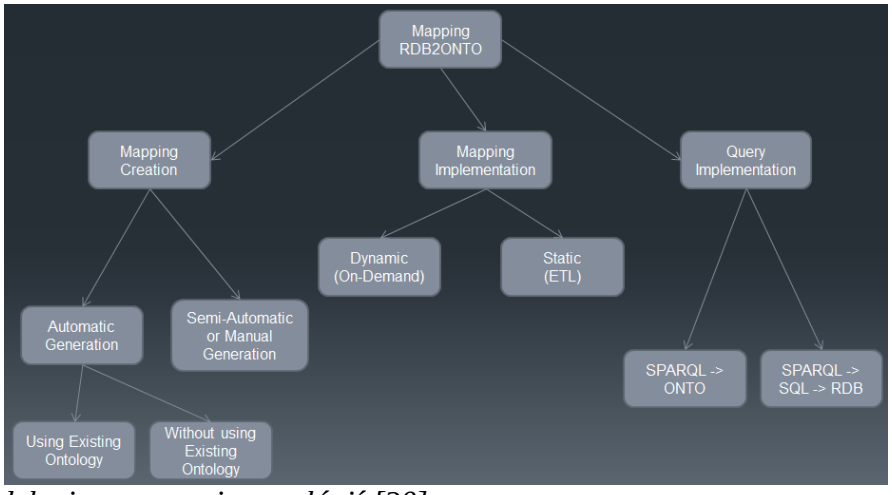
Obr. 16 *Ukážka nástroja RDBAnalyzer.*

2.9 Mapovanie relačných databáz do ontológií

Mapovanie RDB do ontológií je jedna z kľúčových metód tejto práce. Ako som už spomínal, hlavným problémom pri orientovaní sa v RDB, ktorú nepoznáme, je jej nedostatočná sémantika. Vytvorením ontológie nad RDB by sme tento problém vyriešili, lenže vyžadovalo to by množstvo ľudskej práce doménové experta (databázový špecialista, ktorý sa v konkrétnej DB vyzná). Iným, ale bohužiaľ iba čiastočným riešením je automatické alebo poloautomatické mapovanie RDB do ontológií [30-35]. Existujú viaceré dôvody na tento proces:

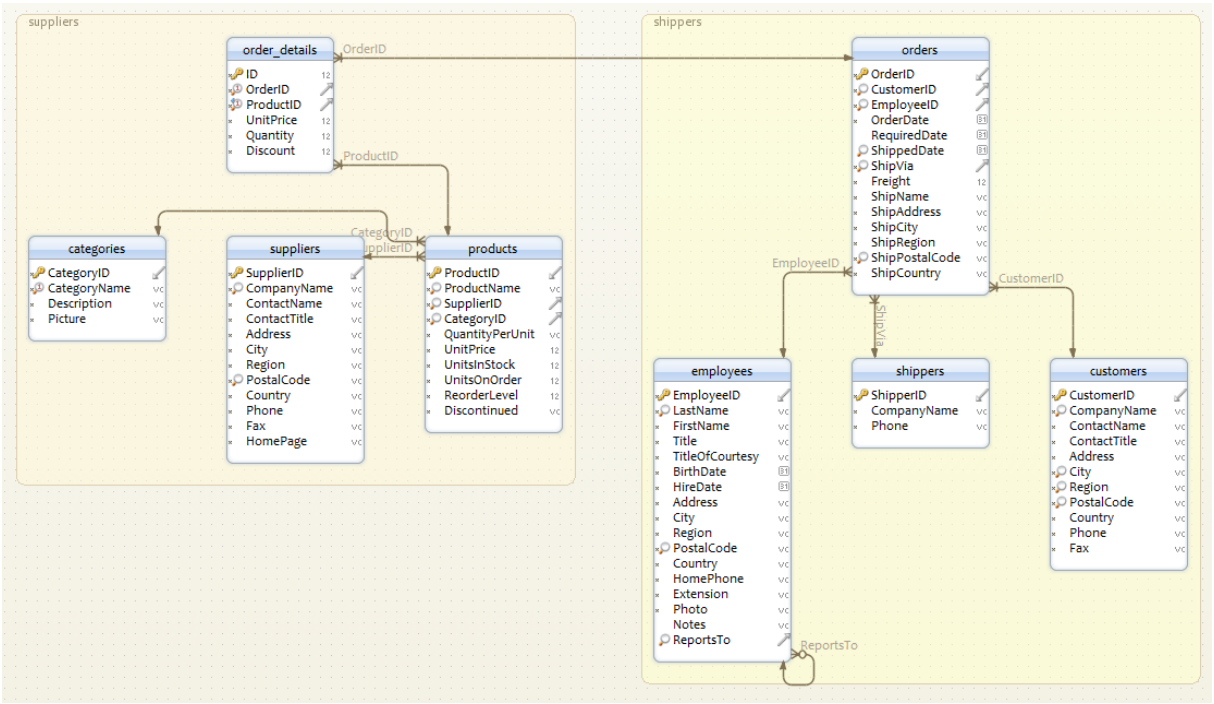
- umožnenie hľadania informácií v RDB bez poznania modelu,
- táto metóda umožňuje strojové vyhľadávanie v RDB(napr. internetové vyhľadávače),
- získavanie informácií z RDB aj ľuďmi, ktorí nepoznajú technológiu RDB (laikom),
- čiastočne sa dajú spájať rôzne RDB, keďže sú metódy na spájanie ontológií, dá sa takto riešiť problém federovaných RDB

Metód mapovania RDB je viacero. Môžeme ich rozdeliť do troch skupín podľa toho, ako mapovanie vytvárame, implementujeme alebo ako implementujeme dotazy nad mapovanými ontológiami (obr. 17). Metóda mapovania závisí najmä od účelu na aký chceme výslednú ontológiu použiť.



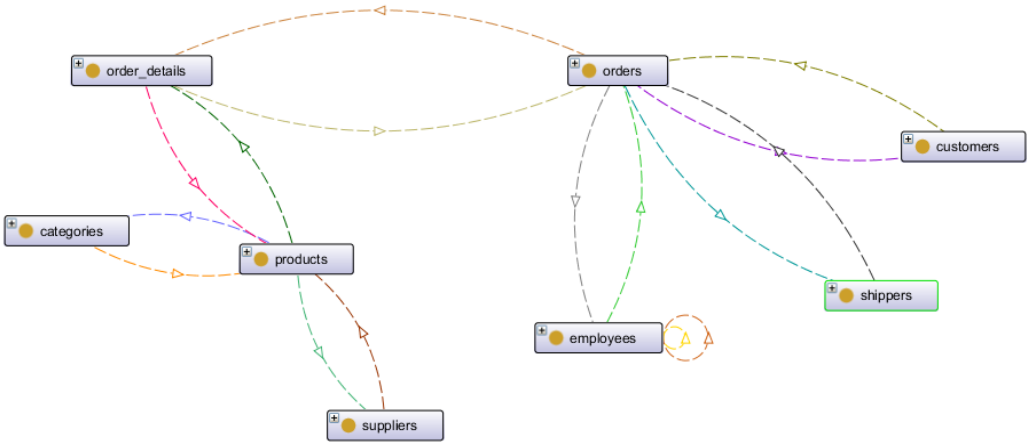
Obr. 17 *Rozdelenie mapovania ontológií [30].*

Pre moju prácu som si vybral statické, plne automatické mapovanie bez použitia existujúcej ontológie. Tento spôsob mapovania je rýchly, nevyžaduje doménového experta a ukážeme, že výsledky sú postačujúce pre účely jeho použitia. Proces generovania prebieha tak, že všetky tabuľky RDB budú triedy, ich atribútmi budú stĺpce s konkrétnymi dátovými typmi, riadky v tabuľkách budú inštancie tried a vzťahy medzi triedami, ale aj inštanciami budú vlastnosti (object property). Pri testovaní som použil softvér RDBToOnto 1.2 Beta. Je určený pre stredne veľké RDB (rádovo 10ky tisíc riadkov v tabuľkách), podporuje MySQL, ORACLE, MS Access, Excel. Výstupná ontológia je vo formáte OWL. Na testovanie som použil menšiu RDB Northwind (voľne dostupná testovacia RDB od firmy Microsoft). Na obrázku 18 si môžeme pozrieť jej schému.



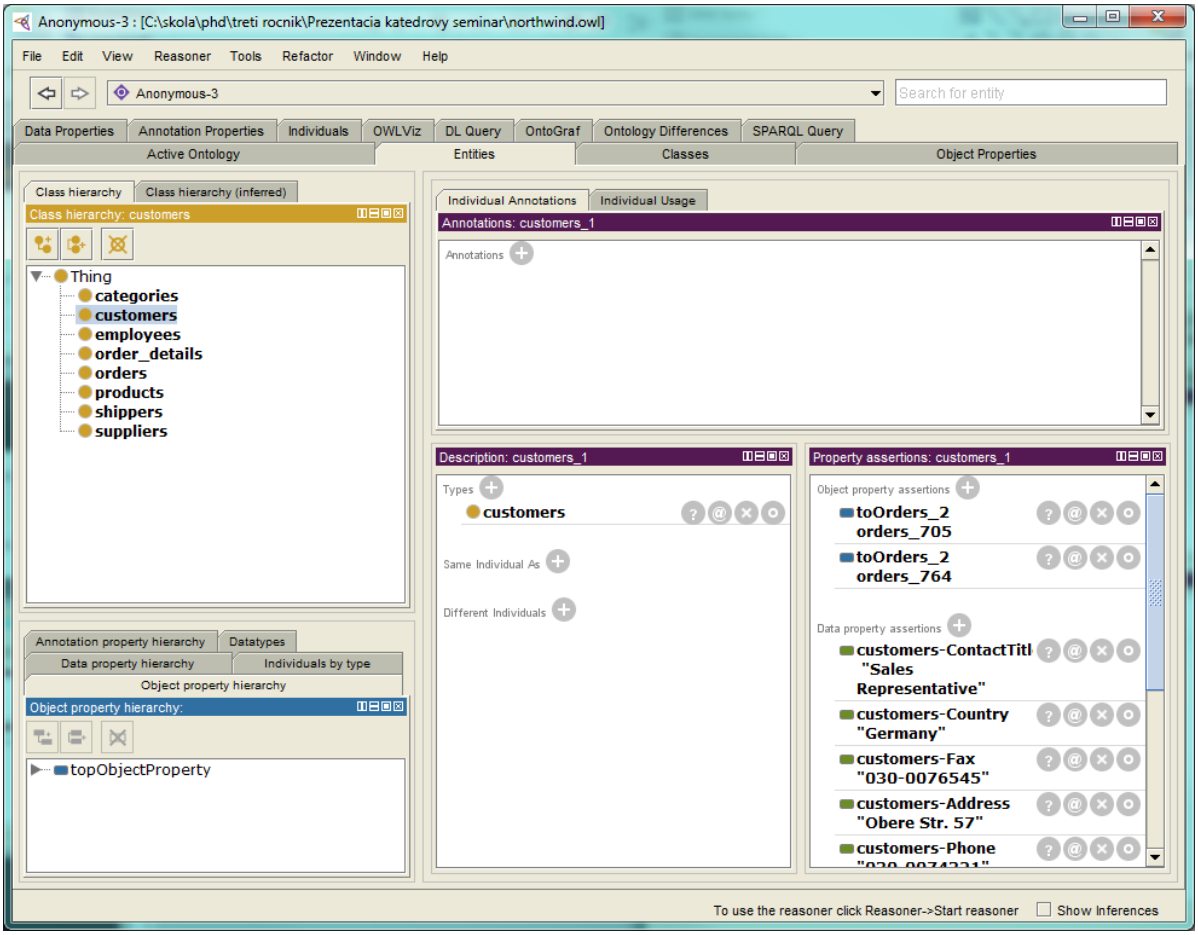
Obr. 18 *Entitno relačný diagram testovacej relačnej databázy NORTHWIND od Microsoftu.*

Po vygenerovaní a zobrazení ontológie vo voľne dostupnom nástroji Protégé 4.2 Beta z nej môžeme vidieť graf tried (Obr 19). Vyzerá podobne ako ER diagram na obr. 18 avšak vzťahy sú všade obojstranné. To je spôsobné tým, že v ontológiách sú vždy aj inverzné vzťahy. Nástroj Protége nám umožnuje aj interaktívne vo vizualizácií zobrazovať atribúty tried, inštancie, ich konkrétne hodnoty a aj konkrétne vzťahy medzi nimi. Ďalej je možné pomocou neho aj vyhľadávať objekty pomocou textového vyhľadávača, dotazovať sa pomocou jazyka SPARQL, ale tiež aj odvodzovať ďalšie znalosti z ontológie pomocou algoritmov FaCT++ a HermiT. Pri takto triviálne namapovanej RDB nám však odvodzovače veľa toho neodvodia. Na odvodenie použiteľných znalostí by musela byť ontológia dotvorená doménovým expertom.



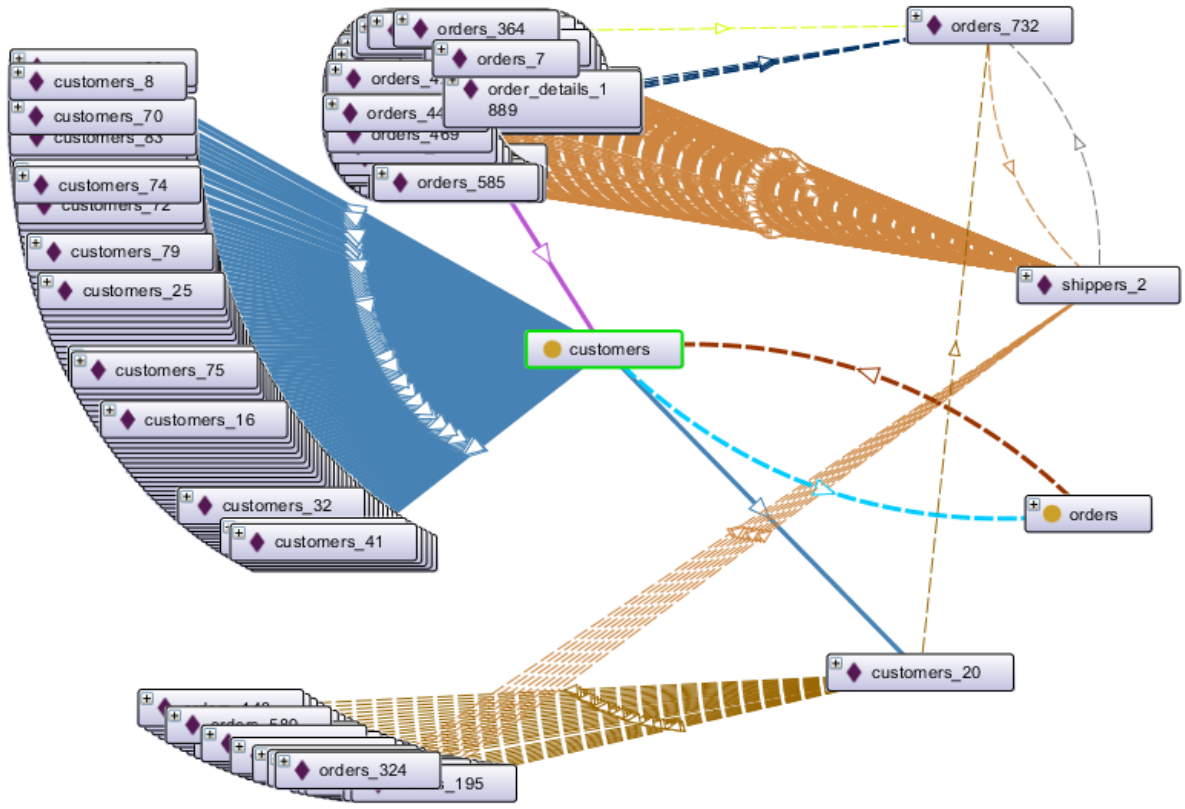
Obr. 19 *Vizualizovanie tried v namapovanej ontológie pomocou aplikácie Protége. Triedy sa dajú pomocou nástroja interaktívne rozkliknúť a hľadať v nich informácie.*

Na ukážke (Obr. 20) je zobrazený interfejs nástroja Protége a spôsob akým prehľadávame ontológiu, vygenerovanú z databázy. Jednoducho môžeme prehľadávať triedy a inštancie a pátrať po informáciách, ktoré nás zaujímajú. Konkrétne na tejto ukážke máme označenú triedu zákazníkov (customers), vpravo dole zelenou farbou vidíme atribúty jedného zo zákazníkov, ktorého sme si vybrali a nad nimi sú modrou vzťahy ku konkrétnym inštanciám triedy objednávky (orders).



Obr. 20 *Ukážka nástroja Protége, vľavo hore sú triedy, po rozkliknutí vidíme ich inštancie (vpravo dole) a vzťahy (vpravo v strede).*

Jednoducho môžeme kliknúť na konkrétnu objednávku a pozrieť si jej detaily prípadne iné informácie, ktoré so zákazníkom súvisia. Pomocou vizualizácie v tomto nástroji si môžeme zobraziť aj ich inštancie a ich prepojenie (obr. 21). Na ukážke vidíme napríklad, že zákazník s id 20 (na obr. 21 vpravo dole) má vzťah s triedou customers (je jej inštancia, hrubá modrá čiara). Má objednávky, slabohnedé prerušované čiary. Jeho konkrétna objednávka 732 (na obr. 21 v pravo hore) je od dodávateľa 2 a podobne. Protége umožňuje tento graf interaktívne prehľadávať.



Obr. 21 *Vizualizácia vzťahov medzi inštanciami a triedami.*

Takýmto spôsobom dokáže aj laik hľadať informácie, nielen o obsahu a popise RDB na aké účely slúži, ale dokonca aj vyhľadávať informácie v nej. Tento typ mapovania, ktorý som použil je plne automatický a nevyžaduje si žiadnu ľudskú prácu. Je však veľmi dôležité, aby boli názvy objektov vhodne pomenované, aby sa dali chápať bez dodatočných informácií. To už však závisí od autora RDB, ktorý ju modeloval. V zásade však s tým problém nie je. Treba spomenúť aj nedostatky mapovania RDB do ontológií. Medzi ne patria hlavne nasledovné:

- Je dosť obtiažné udržiavať ontológiu aktuálnu, pri statickej implementácií, by ju bolo treba pravidelne znovu generovať, čo je dosť výpočtovo náročne, keďže RDB obsahujú veľa dát. Pri dynamickej implementácií je zase práca s ontológiou pomalá.
- Pri dotazovaní sa na ontológiu, napr. pomocou jazyka SPARQL, bude čas výpočtu vždy dlhší ako pri použití jazyka SQL nad RDB pri rovnakom dotaze. Často krát by pri agregovaných dotazoch výpočet ani neskončil v reálnom čase.
- Existujúce nástroje, či už na mapovanie RDB, ale taktiež aj na používanie ontológií nedokážu pracovať s rozsiahlymi RDB.

Napriek týmto nedostatkom je toto riešenie dostačujúce na hľadanie informácií a znalostí v malých a stredne veľkých databázach. Na použitie pri veľkých a rozsiahlych RDB by sme

však potrebovali upraviť alebo vytvoriť efektívne implementácie jednak na mapovanie RDB (pri statickom by to nemal byť problém), ale tiež nájsť vhodný nástroj na zobrazenie a hľadanie v tak veľkej ontológií aká by sa vygenerovala. Použitý nástroj Protége mal problém s pamäťou už pri stredne veľkých ontológiách. Ďalším dobrým riešením by bolo generovanie ontológie iba zo schémy RDB bez záznamov, kde by sa dynamicky pri zobrazovaní inštancií volali SQL dotazy na RDB. Užívateľ by mal náhľad k ontológií so sémantikou štruktúry alebo modelu RDB a interaktívne pri požiadavkách by sa automaticky generoval SQL dotaz, ktorý by mu zobrazil informácie o inštanciách v ontológií.
Myšlienka polo/automatického sémantického popisu RDB nie je nová. Objavil som ju dokonca v článku z roku 1979 [31], v období keď sa pojem ontológie v informatike ani nepoužíval. Autori článku navrhli urobiť z tabuliek a vzťahov v RDB graf, kde pomenovali hrany výstižnými pojmami, aby bolo z nich jasné ako prepájajú tabuľky (vrcholy grafu). Potom sa v RDB vyhľadávalo tak, že sa interaktívne chodilo po grafe a z pochopenia popisov a spájaním vrcholov sa tvorili dotazy. Nepodarilo sa mi však zistiť ako ich výskum pokračoval, či to nejako vyvíjali a prečo sa to teraz nepoužíva.

3 Záver

V práci som sa venoval problematike spracovania, hľadania informácií a orientácie v rozsiahlych, najmä relačných databázach. V teoretickej časti som popísal zdroje a formy dát, spôsoby uchovávania týchto dát v rôznych databázových systémoch. Podrobne sú v práci popísané relačné databázové systémy a spracovanie dát v nich. Vysvetlený je problém rozsiahlych databáz, ktoré sú schopné technologicky uchovávať a spracovávať veľké množstvo dát, avšak pre človeka sú ťažko pochopiteľné a bez vývoja nových metód a algoritmov nie je možné plne využiť potenciál informácií, ktoré obsahujú. Zameral som sa najmä na metódy a postupy, ktoré slúžia pri orientácií a snahe pochopiť rozsiahlu relačnú databázu za účelom vyhľadania dôležitých informácií v nej, jej analýzy, prípadne rozšírenia jej dátového modelu databázovým architektom. V práci je ďalej popísaná vizualizácia dát a databáz, databázové metriky, ontológie, teória komplexných sieti a grafov, čo sú kľúčové východiská k riešeniu tohto problému v tejto práci. V práci som vylepšil a implementoval vizualizáciu SchemaBall (2.2.1), ktorá slúži na zobrazenie entitno relačného diagramu a zvýraznenie dôležitých častí v relačnej databáze so zložitým dátovým modelom. Navrhol a popísal som praktický postup ako skúmať rozsiahlu, zložitú alebo neznámu databázu a pochopiť ju tak, aby sa dal chápať jej účel a dali hľadať v nej informácie (2.5). Ukázal som, že pri zväčšovaní dátového modelu relačnej databázy má jej graf (vzťahy a tabuľky) charakteristiky bezškálovej siete, čo sa dá využiť pri jej skúmaní a orientovaní sa v nej (2.6). Ďalej rôzne prístupy transformácie grafu databázy a jej skúmanie pomocou vizualizácie grafov (Gephi) a grafových algoritmov.

Všetky tieto prístupy a metódy som implementoval v otvorenom, ľahko rozšíriteľnom nástroji RDBAnalyzer (2.8), ktorý slúži na orientáciu a analýzu rozsiahlych relačných databáz. Metódy ako aj nástroj RDBAnalyzer boli overené na reálnej rozsiahlej databáze CEHZ a ďalších dvoch reálnych databázach. Ďalej som ukázal možnosti ako transformovať relačné databázy do ontológií a navrhol postup ako sa v takto vytvorených ontológiách orientovať a vyhľadávať v nich informácie (2.9). Týmto spôsobom je možné vyhľadávať a sprístupniť informácie z relačných databáz aj ľuďom, ktorí neovládajú túto technológiu, strojom alebo webovým vyhľadávačom. Tento prístup sa dá však momentálne použiť len na malé a stredne veľké relačné databázy.

3.1 Vedecký prínos práce

Vedecký prínos mojej práce by som zhrnul do nasledovných bodov:

1. Návrh a tvorba metódy na orientáciu, pochopenie a hľadanie informácii v rozsiahlych alebo neznámych relačných databázach (2.5). Vytvorenie nástroja RDBAnalyzer, kde sú spomínané postupy implementované.
2. Zistenie, že grafy schém a dát rozsiahlych relačných databáz majú charakteristiky bezškálových sietí (2.6) a využitie tohto poznatku (overenie a návrh databázových metrík).
3. Návrh a tvorba metódy pre orientáciu a hľadanie informácií v ontológiách generovaných z relačných databáz (2.9).

3.2 Možnosti ďalšieho vývoja a výskumu

Na téme by sa dalo ďalej pracovať hlavne pri probléme mapovania relačných databáz do ontológií (2.9). Súčasné riešenia sú postačujúce tak na malé a stredne veľké databázy. S implementáciou pre veľké databázy som sa nestretol. Navyše ak by aj existovala táto implementácia tohto mapovania, nebolo by ju zložité urobiť na základe programu, ktorý som vytvoril na vygenerovanie grafu n-tíc z rozsiahlych RDB (2.6.2 Prístup 3), tak ďalším problémom by bolo, v akom nástroji na spracovanie ontológií by sme ju chceli používať. Pre všetky dostupné nástroje by bola vygenerovaná ontológia priveľká. Vyžadovalo by to teda vytvoriť nástroj na editovanie a hľadanie informácií takto veľkých ontológií. Mohol by to byť nový modul nástroja RDBAnalyzer. Alebo tiež urobiť nejakú rozumnú kombináciu, kde by napríklad ontológia obsahovala nejaké časti databázy (tabuľky a vzťahy) a ďalšie (záznamy) by sa dali interaktívne dotazovať z aktívnej databázy a podobne. Navyše automatické mapovanie, ktoré som v práci popísal a overil, by sa mohlo vylepšiť o ďalšie sémantické informácie z databázy.

Ďalšou možnosťou vylepšenia v práci popísaných metód a vizualizácii na orientáciu a hľadanie dôležitých častí v rozsiahlej databáze by mohlo byť aj zapracovanie ďalších metainformácií do metrík a rankingov. Informácie o používaní a zmenách jednotlivých tabuliek, záznamov, objektov v databáze a záznamy z logov nám môžu veľa hovoriť o dátach v databáze a ich dôležitosti.

4 Zoznam Použitej Literatúry

Vlastné publikácie

[1] Takac L., *Data Processing Over Very Large Databases,* Winter School MICT, Šachtičky Slovakia, 3-8 Jan. 2011.

[2] Takac L., *Visualization of Large Multivariate Data Sets using Parallel Coordinates,* Transcom, Žilina Slovakia, Jún. 2011.

[3] Takac L., *Data Processing and Analyzing over Very Large Relational Databases,* MICT 2012, 7-th Winter School of Mathematics Applied to ICT, Šachtičky, feb. 2012.

[4] Takac L., Zabovsky M., *Data Analysis in Public Social Network,* International Scientific Conference & International Workshop Present Day Trends of Innovations, 2012.

[5] Takac L., Zabovsky M., *Design and development of new automatic on-line media monitoring system*, 7th International Conference on Telecommunication Systems, Services, and Applications (TSSA), 2012.
[6] Takac L., *An Approach for Orientation over very large Relational Databases*, Vyssheye tekhnicheskoye obrazovaniye:problemy i puti razvitiya, ISBN: 978- 985-488-934-4, Minsk, 2012.
[7] Takac L., *Retrieving Data From Public Social Networks Based on Scale-free Characteristic*, Acta Electrotechnica et Informatica, No. 4, Vol. 12, 2012.
[8] Takac L., Zabovsky M., Matiasko K., MNSight – *A new Automatic on-line Media Monitoring System*, Radiomatics - Journal on Communication Engineering, Vol. 3, No. 2, 2013.
[9] Takac L., Zabovsky M., *Large Personal Data Sets Exploration*, Internet in the information Society, Dabrowa Gornica, Apríl, 2013.
[10] Takac L., *Fast Exact String Pattern-Matching Algorithm for Fixed Length Patterns*, Transcom, Žilina Slovakia, Jún. 2013.

Ostatné zdroje

[11] Coral Calero, Mario Piattini, Marcela Genero, *A Case Study with Relational Database Metrics*, Computer Systems and Applications, ACS/IEEE International Conference 2001.
[12] Serrano M.A., Calero C., Houari, Sahraoui A., Piattini M., *Empirical studies to assess the understandability of data warehouse schemas using structural metrics*, Software Quality Journal, Volume 16, 2008.
[13] Díaz O., Piattini M., *Metrics for Active Database Maintainability*, Advanced Information Systems Engineering Lecture Notes in Computer Science, Volume 1626, 1999.
[14] Piattini M., Calerno C., Genero M., *Table Oriented Metrics for Relational Databases*, Software Quality Journal, Volume 9, 2001.
[15] Calero C., Houari A. Sahraoui, Piattini M., Lounis H., *Estimating Object-Relational Database Understandability Using Structural Metrics*, Database and Expert Systems Applications Lecture in Computer Science, Volume 2113. 2001.
[16] Zábovský, M., Zábovská, K., *Big data,* Proceedings of 12th International Conference System Integration 2010, 6-7 Sept. 2010 .
[17] Chambers J.M, Cleveland W. S, Kleiner B, and Tukey P.A., *Graphical Methods for Data Analysis*, Chapman & Hall, 1983.
[18] Blanchard J, Guillet F, Briand H., *Exploratory Visualization for Association Rule Rummaging*, KDD-03 Workshop on Multimedia Data Mining, 2003.
[19] Ahmed A., Dywer T., Hong S-H., Murray C., Song L., Wu Y.X., *Visualisation and Analysis of Large and Complex Scale-free Networks*. Proceedings Eurographics/IEEE VGTC Symposium on Visualization (EuroVis 2005), 2005.
[20] Dasu T., Johnson T., Muthukrishnan S., Shkapenyuk V., Marathe A., *Bellman: A Data Quality Browser*, International Conference on Information Quality (IQ2001), 2001.
[21] Huotary J., *Introduction to information visualization – tools and techniques*, Žilina 2009.
[22] Krzywinski, M., *Schemaball: A New Spin on Database Visualization*, Sysadmin Magazine Vol 13 Issue 08, 2004.
[23] TRICAUD S., KARA N., SADDÉ P., *Visualizing network activity using parallel coordinates*, System Sciences (HICSS), 44th Hawaii International Conference, 2011.

[24] Johnson C., *Top Scientific Visualization Research Problems*, IEEE Computer Graphics and Applications, vol. 24, no. 4, pp. 13-17, July/Aug. 2004.
[25] Wijk J.J., *The Value of Visualization*. Proc. IEEE Conf. Visualization, 2005.
[26] Eick S.G., *Information Visualization*, 10. IEEE Computer Graphics and Applications, vol. 25, no. 1, pp. 12-14, Jan./Feb. 2005.
[27] Wijk J.J., *Views on Visualization*. IEEE Transactions on Visualization and Computer Graphics, vol. 12, no. 4, pp. 421-433, July/Aug. 2006.
[28] Barabási A. L., Bonabeau E., *Scale-Free Networks*, Scientific American, MAY 2003.
[29] SNAP, http://snap.stanford.edu/, 2013.
[30] Sahoo S.S., Halb W., Hellmann S., Idehen K., Thibodeau T., Auer S., Sequeda J., Ezzat A., *A Survey of Current Approaches for Mapping of Relational Databases to RDF*, W3C Incubator Group, 2009.
[31] Kashyap R.L., Abhyankar R.B., *Semantics for data retrieval in relational database systems,* Computer Software and Applications Conference, Proceedings. COMPSAC 79. The IEEE Computer Society's Third International , vol., no., pp. 319- 324, 1979.
[32] Gherabi N., Addakiri K., Bahaj .M., *Discovering new technique for mapping relational database based on semantic web technology*, The 2nd International Conference on Communications and Information Technology (ICCIT), Hammamet 2012.
[33] Chen A., Liu L., Shang J., *A hybrid strategy to construct scientific instrument ontology from relational database model*, International Conference on Computer Distributed Control and Intelligent Enviromental Monitoring, 2012.
[34] Fatima S., Rajput Q., *A Comparison of Algorithms to Construct Ontologies from Relational Databases*, The Seventh International Multi-Conference on Computing in the Global Information Technology (ICCGI), 2012.
[35] Zárate J.A.E., Espinosa A.T., Rosas R.M.V., Sanchez L.M., *Semantic Classification of Attributes for Integrating Heterogeneous Relational Databases with Artificial Neural Networks*, Electrical Engineering Computing Science and Automatic Control (CCE), 2011 8th International Conference on , vol., no., pp.1-5, 26-28 Oct. 2011.
[36] GEPHI, https://gephi.org/,2013.